150 ptas.
51
mi computer
CURSO PRACTICO DEL ORDENADOR PERSONAL,
EL MICRO Y EL MINIORDENADOR
Editorial Delta, S.A.

MI COMPUTER CURSO PRACTICO
DEL ORDENADOR PERSONAL, EL MICRO Y EL MINIORDENADOR

Publicado por Editorial Delta, S.A., Barcelona

Volumen V - Fascículo 51

Director: José Mas Godayol
Director editorial: Gerardo Romero
Jefe de redacción: Pablo Parra
Coordinación editorial: Jaime Mardones, Francisco Martín
Asesor técnico: Ramón Cervelló

Redactores y colaboradores: G. Jefferson, R. Ford, F. Martín, S. Tarditti, A. Cuevas, F. Blasco
Para la edición inglesa: R. Pawson (editor), D. Tebbutt (consultant editor), C. Cooper (executive editor), D. Whelan (art editor), Bunch Partworks Ltd. (proyecto y realización)

Realización gráfica: Luis F. Balaguer

Redacción y administración:
Paseo de Gracia, 88, 5.°, 08008 Barcelona
Tels. (93) 215 10 32 / (93) 215 10 50 - Télex 97848 EDLTE

MI COMPUTER, *Curso práctico del ordenador personal, el micro y el miniordenador*, se publica en forma de 96 fascículos de aparición semanal, encuadernables en ocho volúmenes. Cada fascículo consta de 20 páginas interiores y sus correspondientes cubiertas. Con el fascículo que completa cada uno de los volúmenes, se ponen a la venta las tapas para su encuadernación.
El editor se reserva el derecho de modificar el precio de venta del fascículo en el transcurso de la obra, si las circunstancias del mercado así lo exigieran.

ISBN: 84-85822-83-8 (fascículo) 84-7598-007-4 (tomo 5)
84-85822-82-X (obra completa)
Depósito Legal: B. 52/1984

Fotocomposición: Tecfa, S.A., Pedro IV, 160, Barcelona-5
Impresión: Cayfosa, Santa Perpètua de Mogoda (Barcelona) 028501
Impreso en España - Printed in Spain - Diciembre 1984

Editorial Delta, S.A., garantiza la publicación de todos los fascículos que componen esta obra.

Distribuye para España: Marco Ibérica, Distribución de Ediciones, S.A., Carretera de Irún, km 13,350. Variante de Fuencarral, 28034 Madrid.
Distribuye para Argentina: Viscontea Distribuidora, S.C.A., La Rioja 1134/56, Buenos Aires.
Distribuye para Colombia: Distribuidoras Unidas, Ltda., Transversal 93, n.° 52-03, Bogotá D.E.
Distribuye para México: Distribuidora Intermex, S.A., Lucio blanco, n.° 435, Col. San Juan Tlihuaca, Azcapotzalco, 02400, México D.F.
Distribuye para Venezuela: Distribuidora Continental, S.A., Edificio Bloque Dearmas, final Avda. San Martín con final Avda. La Paz, Caracas 1010.

Pida a su proveedor habitual que le reserve un ejemplar de MI COMPUTER. Comprando su fascículo todas las semanas y en el mismo quiosco o librería, Vd. conseguirá un servicio más rápido, pues nos permite realizar la distribución a los puntos de venta con la mayor precisión.

Servicio de suscripciones y atrasados (sólo para España)

Las condiciones de suscripción a la obra completa (96 fascículos más las tapas, guardas y transferibles para la confección de los 8 volúmenes) son las siguientes:

a) Un pago único anticipado de 16 690 ptas. o bien 8 pagos trimestrales anticipados y consecutivos de 2 087 ptas. (sin gastos de envío).

b) Los pagos pueden hacerse efectivos mediante ingreso en la cuenta 6.850.277 de la Caja Postal de Ahorros y remitiendo a continuación el resguardo o su fotocopia a Editorial Delta, S.A. (Paseo de Gracia, 88, 5.°, 08008 Barcelona), o también con talón bancario remitido a la misma dirección.

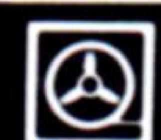
c) Se realizará un envío cada 12 semanas, compuesto de 12 fascículos y las tapas para encuadernarlos.

Los fascículos atrasados pueden adquirirse en el quiosco o librería habitual. También pueden recibirse por correo, con incremento del coste de envío, haciendo llegar su importe a Editorial Delta, S.A., en la forma establecida en el apartado b).

Para cualquier aclaración, telefonear al (93) 215 75 21.

No se efectúan envíos contra reembolso.





Hecho a la medida

La ergonómica es la ciencia que trata de adecuar el ambiente laboral al trabajador. En ella inciden disciplinas tan diversas como la ingeniería y la psicología

La introducción de los ordenadores personales en los entornos doméstico y laboral hace que las consideraciones sobre la ergonómica revistan suma importancia. El color y el sonido constituyen buenos ejemplos respecto a cómo influye el ambiente que rodea a las personas en su rendimiento en el trabajo. Algunas centralitas telefónicas ofrecen música a quienes llaman para relajarlos y tranquilizarlos durante la espera. Una pantalla VDU verde resulta más agradable y relajada a la mayoría de los usuarios de ordenadores, mientras que las pantallas en blanco y negro producen cansancio y tensión.

Diferentes combinaciones de colores provocan respuestas muy distintas: azul sobre amarillo por lo general se considera una visualización atractiva, mientras que cyan (azul claro) sobre verde no suele agradar. El color de una habitación puede influir en el estado de ánimo de la persona que esté en ella: los amarillos se consideran "alegres", los azules y verdes "relajantes" y el esquema de una oficina típica, marrón y gris, "deprimente".

Las sutiles diferencias de este tipo en cuanto a preferencia podrían parecerle irrelevantes o intrascendentes al usuario de un micro personal, pero los especialistas en ergonómica han demostrado repetidamente que cuando las personas se sienten insatisfechas con su entorno (con frecuencia sin saber que es el esquema de color o el ruido de fondo lo que les está afectando), son más propensas a la fatiga visual y el dolor de espalda. Los usuarios de ordenadores sufren comúnmente las consecuencias de la disposición irracional de las habitaciones y las estructuras de trabajo. Un importante estudio ergonómico sobre una oficina en la cual los operadores trabajaban en terminales acomodados junto a las paredes, atribuyó el bajo rendimiento y el elevado porcentaje de errores al aislamiento social y la tensión producidos por la distribución. Se descubrió que cuando los trabajadores se colocaban frente a frente en pantallas de perfil bajo y en medio de la habitación, toda la atmósfera se avivaba y la calidad del trabajo mejoraba.

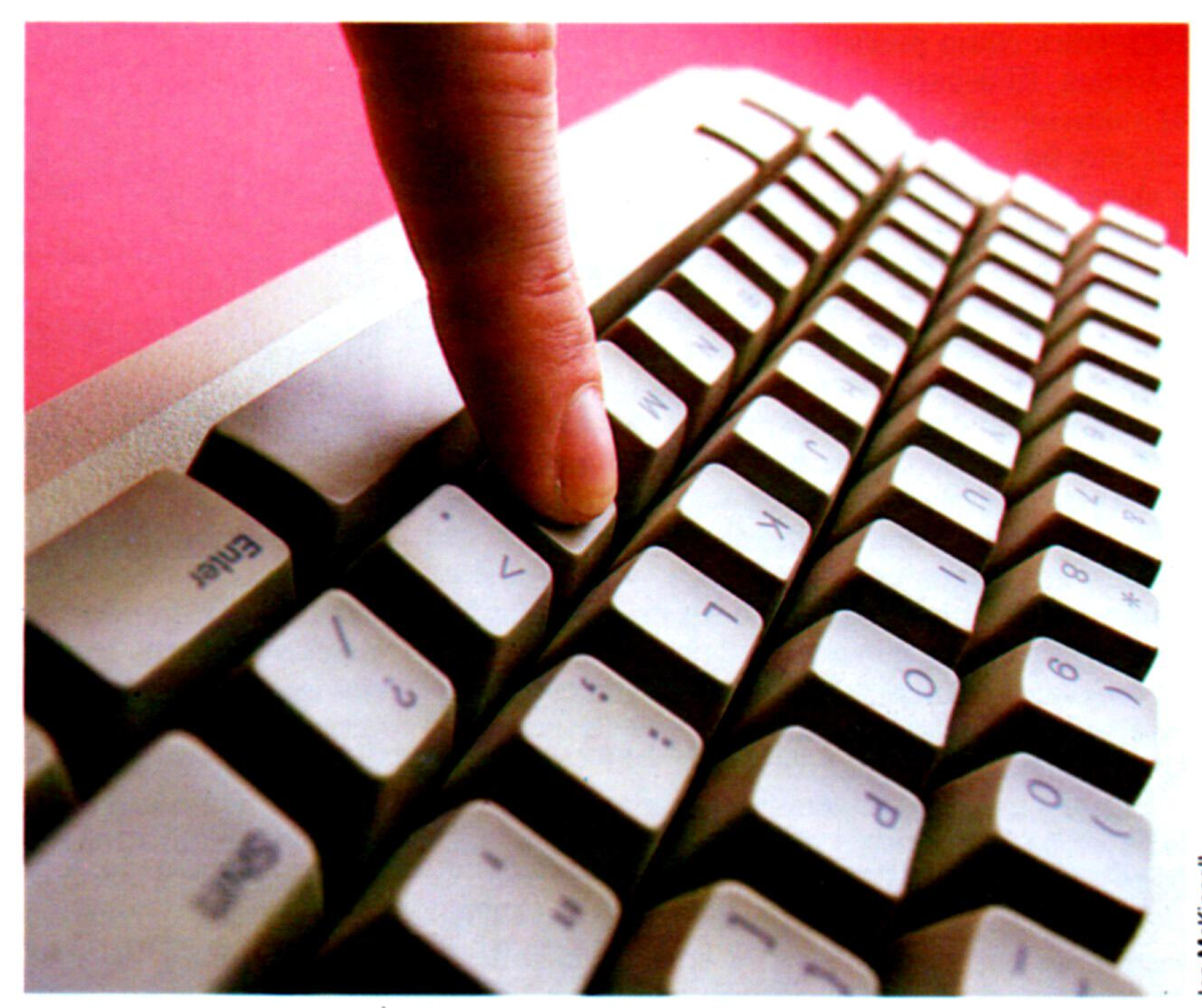
La introducción de los ordenadores frecuentemente ha tenido un efecto de "pérdida de experiencia" en los trabajos, haciendo poco exigente, rutinario y carente de interés el trabajo de las personas. Ahora ya se reconoce como parte de la misión del analista de sistemas el asegurar que las tareas asignadas a las personas después de la instalación de un ordenador sean satisfactorias y suficientemente exigentes. Para este análisis se necesita el consejo del especialista en ergonómica.

Un diseño de sistema verdaderamente ergonómico considera al operador humano como un importante componente del mismo, con sus propias características operativas y sus tolerancias. Tanto las consideraciones humanitarias como la lógica financiera reconocen que los sentimientos, las percepciones y los hábitos de las personas son tan importantes para la eficacia de un sistema como la capacidad del bus de direcciones o la velocidad del reloj.

El programador también se puede beneficiar del estudio atento de los problemas y las exigencias particulares de su trabajo. La programación exige una atención constante, un método lógico y un concienzudo cuidado al detalle, pero son pocas las personas que exhiben naturalmente estas cualidades y la mayoría de ellas se resiste a su imposición. Ésta es una de las razones por las cuales es imposible garantizar programas libres de errores.

La aplicación de la ergonómica al diseño de nuevos lenguajes de programación y a métodos para el diseño de sistemas es una técnica fascinante que ofrece muchos beneficios tanto a los programadores como a quienes los emplean. Los psicólogos se han pasado años estudiando a las personas como dispositivos procesadores de información y tienen ideas claras (aunque aún incompletas) acerca de las estructuras de memoria, velocidades y capacidades del cerebro. Ellos pueden ayudar a diseñar estructuras de datos y de control de programas con las que las personas se sientan cómodas y las puedan utilizar naturalmente o con escaso entrenamiento.

Las consideraciones psicológicas son, asimismo, una parte de los métodos de entrenamiento, selección y organización dentro de la industria. Más im-



Ian McKinnell

El primer contacto
En general, la interface para el usuario empieza en el teclado, que ha sido objeto de muchas mejoras ergonómicas, desde las teclas esculpidas y los teclados cóncavos hasta los teclados numéricos y las visualizaciones LCD. Pero el principal defecto del teclado, el de la ineficacia del trazado QWERTY, parece imposible de erradicar, porque la mayoría de las personas se muestran reacias a aprender nuevas pautas de mecanografía. Las emociones típicamente humanas de esta naturaleza y la aparente irracionalidad con frecuencia constituyen los mayores obstáculos con que se topa la ingeniería de los factores humanos

portante aún, los psicólogos pueden asegurar que los diseñadores se concentren primero en hacer que el sistema se adapte a las personas, en vez de dar por sentado que será el usuario quien se haya de adaptar al sistema.

En los últimos años los especialistas en ergonómica han estado estudiando la reacción de las personas ante el software complejo, lo que se ha dado en llamar la *interface para el usuario*. Hasta hace poco, los usuarios de ordenadores han sido profesionales experimentados y sumamente motivados, preparados para aceptar la incomodidad y las molestias y deseosos de adquirir un nivel de competencia técnica como precio a pagar por el poder de la máquina. El usuario de hoy en día, sin embargo, es literalmente una persona común, con una tolerancia muy limitada hacia máquinas temperamentales y exigentes; si tuviera que aprenderse un lenguaje de comandos para bases de datos para poder utilizar un cajero automático, ¡probablemente se llevaría sus negocios a las empresas financieras! Y los usuarios que tienen problemas de interface no son casos aislados. Los sistemas de bases de datos que ponen megabytes de información de administración en cada uno de los escritorios de una oficina están poco y mal utilizados en todas partes, porque para extraer información de ellos se requiere cierta fluidez en lenguajes complicados, como el SQL. Tanto los investigadores como los usuarios confían en que la próxima generación de sistemas *front end* de lenguajes naturales haga que sea posible comunicarse con la base de datos o el modelo financiero en un lenguaje humano corriente.

Existen muchas maneras de ayudar al usuario, incluyendo la provisión de facilidades de "ayuda" (*help*) (véase p. 1006). Las investigaciones en el campo de la inteligencia artificial han llevado al desarrollo de programas basados en el conocimiento que pueden explicar cómo llegan a las decisiones, y se cree que estas técnicas podrían ayudar a la creación de "módulos asesores" para guiar a los usuarios de programas. Este tipo de software "inteligente" plantea, no obstante, una cuestión filosófica: ¿qué es una buena explicación, y para quién? El programador es posible que espere una explicación de las estructuras de datos y los procesos involucrados en un sistema, mientras que el usuario de gestión quizá prefiera un análisis de los medios para alcanzar un fin comercial. Las distinciones de esta clase constituyen problemas semánticos y lingüísticos para los especialistas en ergonómica y los científicos de la informática.

Existe un área, igualmente confusa, del estudio de la interface para el usuario a la que los psicólogos llaman *elaboración de modelos*. Las personas utilizan modelos mentales (tales como estereotipos nacionales o racionales) como forma de rellenar vacíos de su conocimiento; la mayoría de nosotros podríamos predecir los puntos de vista, la personalidad y las ideas políticas de un extraño simplemente a partir del conocimiento del trabajo que realiza. Del mismo modo, las personas enfocan los ordenadores con ideas estereotipadas acerca del poder y el comportamiento de la máquina, y pueden considerarlos más inteligentes y mejor informados que seres humanos que realicen la misma clase de tareas. Cuando se encuentran con el tipo de respuestas lacónicas e impersonales que efectúan muchos paquetes de software (especialmente ante entradas incorrectas), tienden a considerar que los ordenadores son antipáticos, incluso hostiles, apreciaciones que están totalmente fuera de lugar. Esta percepción personalizada de la máquina influye sobre toda la interacción, llevando a menudo a respuestas inapropiadas desde ambos lados de la consola.

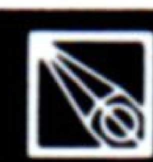
Los programadores que intenten infundir amabilidad hacia el usuario pueden agudizar el problema al introducir características que hagan que el programa parezca más inteligente de lo que en realidad es. Por ejemplo, utilizar avisos graciosos como HOLA JUAN, SOY EL ESPIRITU DE LA BASE DE DATOS puede estimular al usuario a responder de guisa similar, a menudo con resultados catastróficos y el consiguiente desaliento. La genuina amabilidad hacia el usuario necesita de un enfoque más experto. Implica pensar en las personas y preocuparse por ellas, y dejar un margen para sus complicadas reacciones ante los ordenadores y su trabajo.



Condiciones de trabajo
Está generalmente reconocido que el rendimiento de las personas en el trabajo se ve influido por factores físicos manifiestos, tales como la altura de los escritorios, la temperatura ambiental y los niveles de ruido. Sin embargo, efectos más sutiles de color, luz y percepción del espacio actualmente se están reconociendo como aspectos igualmente importantes del sistema de trabajo, en especial cuando se utilizan ordenadores. Un analista que instale un sistema nuevo debe considerar todos estos factores, además de elegir el hardware y el software

El flexible FORTH

Las personas se pueden sentir frustradas y limitadas por ideas recibidas en la misma medida en que por el diseño de su entorno físico. El FORTH, el lenguaje que muchos consideran un sustituto del BASIC, fue inventado por el astrónomo Charles Moore en el Kitts Peak Observatory de Arizona. Moore estaba trabajando en el control de los movimientos del telescopio utilizando programas en FORTRAN, pero ese lenguaje le resultaba demasiado enfocado a procesos de puro cálculo e inadecuado para aplicaciones de control externo. Encontró la solución a su problema escribiendo un nuevo lenguaje. El FORTH se diferencia de los lenguajes rígidamente estructurados, como el FORTRAN, en que permite que el usuario cree virtualmente una nueva versión del lenguaje para adaptarse a cada nueva tarea de programación. El precio de la flexibilidad del FORTH es su enfoque, severamente hostil.



Mecanismo protector

En este capítulo investigaremos formas de producir una salida desde la puerta para el usuario y esbozaremos una amplia red de control

En la mayoría de las aplicaciones de microprocesadores, el término "tampón" *(buffer)* (véase p. 688) ha llegado a significar un lugar de almacenamiento temporal para datos que se están transfiriendo de una parte de un sistema informático a otra. En electrónica analógica, sin embargo, el término se utiliza para describir un circuito que protege a un dispositivo de las acciones de otro. Si deseamos conectar y activar motores eléctricos u otros componentes eléctricos al ordenador para que éste los controle mediante la puerta para el usuario, entonces debemos proteger a los delicados circuitos internos del micro de los componentes a los que lo hemos unido.

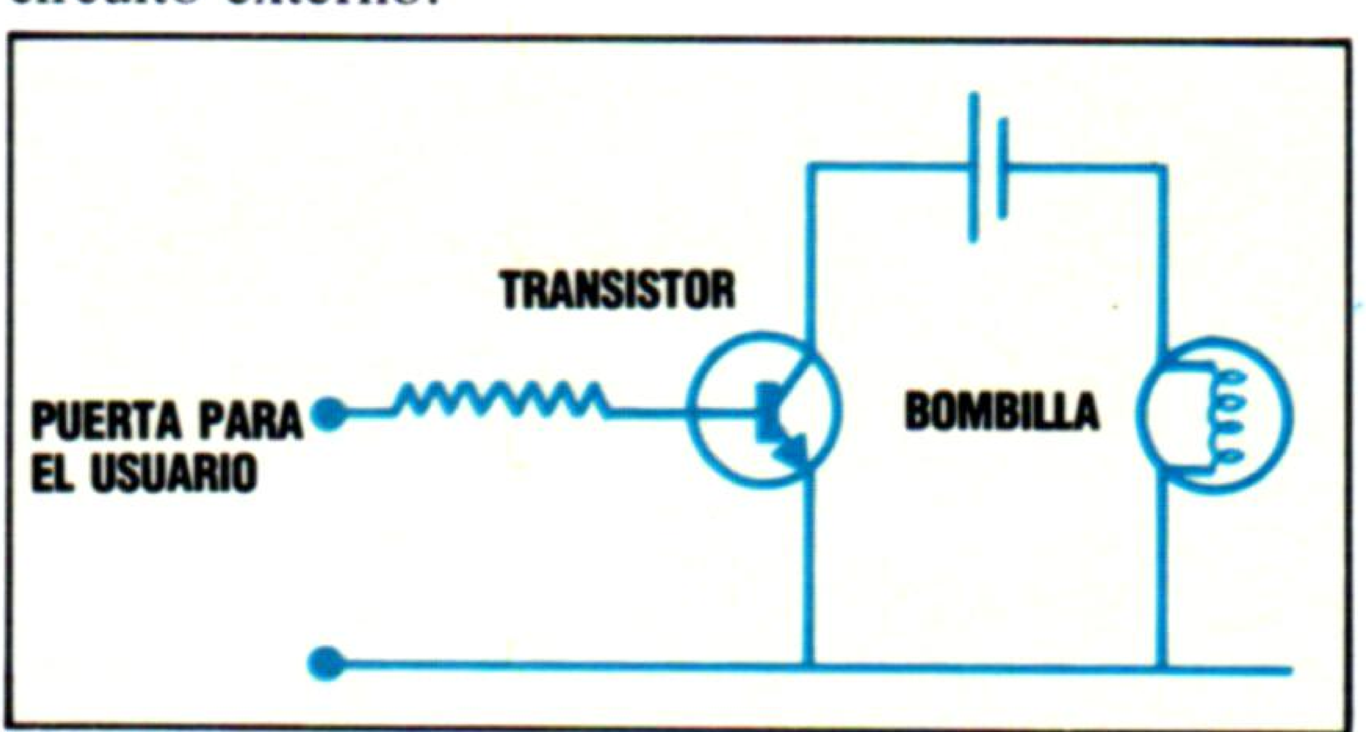
El chip de entrada/salida que hay dentro de su microordenador trabaja a niveles de voltaje de 0 y +5 voltios y utiliza corrientes medidas en unos pocos miliamperios (mA). Por lo tanto, hemos de asegurar que no coloquemos voltajes mayores de +5 voltios en ninguna de las líneas de la puerta para el usuario, ni que usemos más de 30 o 40 mA aproximadamente de corriente.

En nuestro capítulo de introducción de este proyecto (véase p. 994) le mostramos cómo poniendo en contacto entre sí los hilos pelados del cable de la puerta para el usuario se podía modificar el contenido del registro de datos. Descubrimos que conectando a tierra las celdas del registro evitábamos la introducción de voltajes o corrientes peligrosas en el sistema y, por consiguiente, no se requería protección para el sistema de circuitos interno del micro. Sin embargo, si deseamos conectar otros dispositivos debemos incluir una protección y ello se puede hacer de diversas formas. Podríamos desear asegurar solamente que no se usaran más de 20 mA de corriente desde cualquier patilla de la puerta. Esto se puede hacer utilizando un relé conectado a la puerta para el usuario para encender y apagar el dispositivo de salida, e insertando una resistencia en el circuito de alimentación del relé. Si el circuito opera a +5 voltios y nosotros deseamos una corriente de no más de 20 mA, podemos emplear la ley de Ohm (voltaje = corriente × resistencia) para calcular el valor de resistencia necesario:

PUERTA PARA EL USUARIO
RELÉ
MOTOR

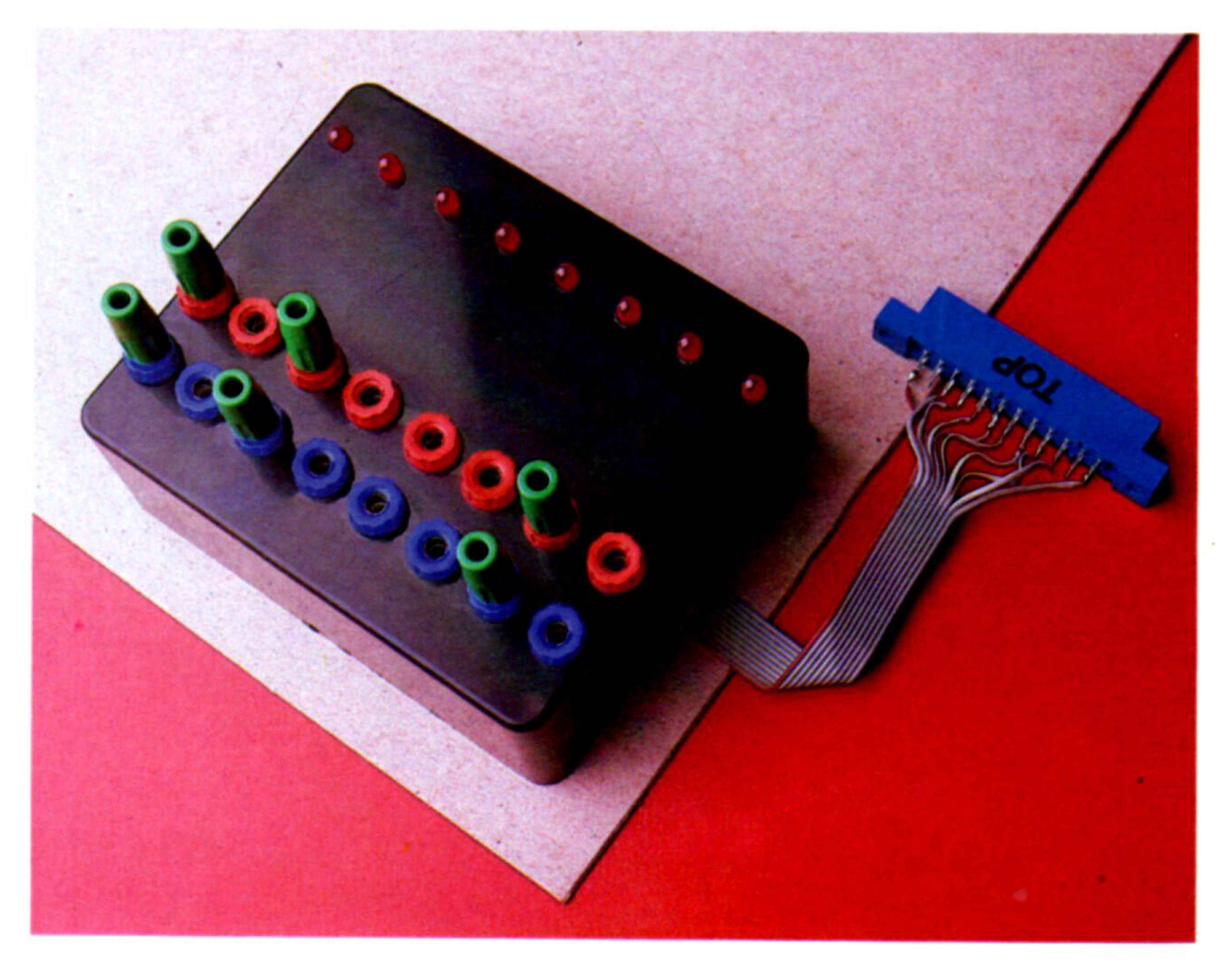
$$V = I \times R$$
$$R = V/I$$
$$R = 5/0{,}02$$
$$R = 250 \text{ ohms}$$

Por otra parte, se podría utilizar la salida de una patilla de la puerta para el usuario para disparar un interruptor de transistor de modo que complete un circuito externo:



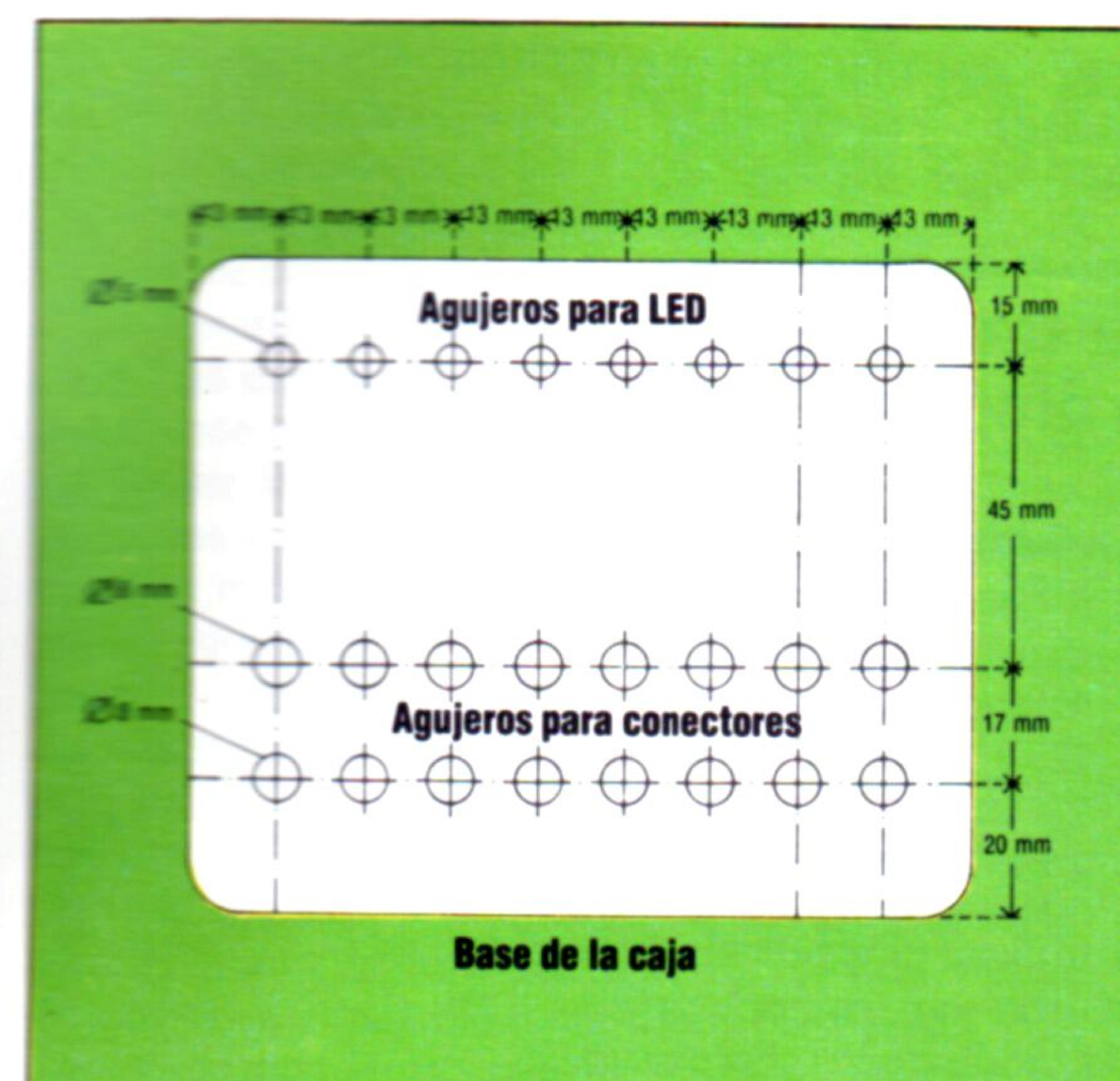
La caja tampón o *buffer* que construiremos utiliza el principio de conmutación del transistor para proteger la puerta para el usuario. Esto es una cuestión de conveniencia, pues los circuitos para las ocho líneas están disponibles todos en un único chip.

Después de conseguido el tamponamiento de la puerta para el usuario podemos seguir adelante para agregarle una serie de módulos que nos permitirán conectar a la misma otros dispositivos. Estos módulos permitirán que controlemos la conmutación de LED, motores de voltajes altos y bajos y relés de la red eléctrica. Entonces seremos capaces



El que va en medio
La caja tampón se conecta a la puerta para el usuario del Commodore 64 o el BBC Micro mediante el cable descrito en el capítulo anterior (véase p. 994). La caja protege al ordenador contra niveles de corriente de entrada/salida peligrosos. Los LED indican el estado de las líneas de salida de la puerta para el usuario, y los enchufes y conectores actúan como interruptores on/off en las líneas de entrada

Liz Dixon



Taladrado de la caja

Los LED y los conectores negros y rojos se montan en la base de la caja, como se ve en la ilustración (pág. 1005). El plástico de la caja es blando y acepta fácilmente la perforación, pero para evitar patinazos y rayaduras cubra su superficie con tiras de cinta adhesiva y después marque los agujeros a taladrar. Marque los agujeros con un punzón o perforación suave y luego taladre: un diámetro de 4 mm para los agujeros de LED, y de 8 mm para los de conector

de controlar dispositivos domésticos tales como luces, grabadoras de cassette, televisores, etc. Además, podemos agregar un convertidor de digital a analógico, que nos permitirá activar visualizaciones de siete segmentos decodificadas. Debido al poco voltaje y salida de corriente de la puerta para el usuario, necesitaremos también una fuente de alimentación eléctrica externa de nueve voltios. A medida que se vaya construyendo cada módulo, se irá conectando a un bus común, a lo largo del cual se encaminarán las ocho líneas de datos, una línea de tierra y una línea de potencia de nueve voltios. De esta forma podemos poner uno encima de otro o interconectar módulos al sistema. Éste es, pues, nuestro plan de acción para los próximos capítulos de este apartado de *Bricolaje*.

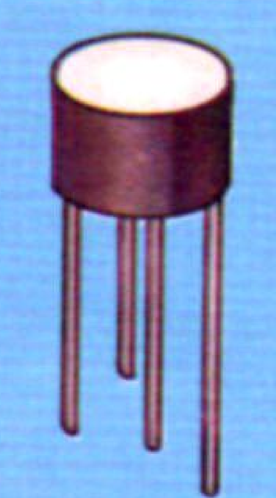
Nomenclatura componentes

Para construir la unidad de interface serán necesarios los componentes que indicamos aquí. Son todos componentes sencillos que se pueden obtener de cualquier proveedor de artículos electrónicos. Las dimensiones exactas de la carcasa no son decisivas, pero nuestro diseño se ajusta exactamente a la caja descrita.

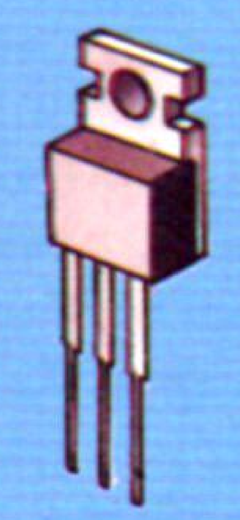
Cantidad	Artículo
8	Resistencia 4,7 K-ohm 0,4 vatios
8	Resistencia 240 ohm 0,4 vatios
1	Condensador electrolítico 1 μF
1	Condensador 0,1 μF
8	LED rojo
8	Diodo 1N4148
1	Puente rectificador W005
3	Buffer hex 7407
1	Regulador de voltaje μA 7805UC
1	Veroboard 36 franjas 50 agujeros
3	Conector chip DIL de 14 patillas
1	Alambre estañado 2 oz/rollo 22
1	Metro cable plano de 12 vías
8	Conector negro 4 mm
8	Conector rojo 4 mm
8	Enchufe negro 4 mm
8	Enchufe rojo 4 mm
1	Conector de potencia 2,1 mm
1	Conector minicon 12 vías
1	Caja plástica 115×95×37 mm

Nuevos componentes

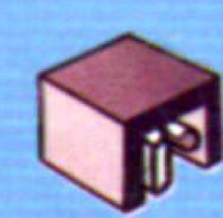
La lista contiene muchos componentes que ya hemos descrito en el curso (véanse pp. 595 y 618), pero los que hemos ilustrado quizá no le resulten familiares. El conector de potencia, el rectificador y el regulador forman parte de la fuente de alimentación eléctrica de la caja. Nuestro diseño nos permitirá utilizar casi cualquier transformador de red eléctrica como entrada, siempre que su salida sea de entre 7 y 25 voltios, CA o CD. El conector de 12 vías se constituirá en la puerta de entrada/salida de la caja a través de la cual los datos pasarán desde y hacia dispositivos externos.



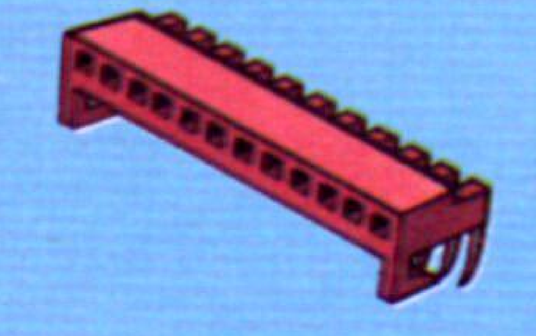
Puente rectificador
Convierte una gama de voltajes de entrada (CA o CD) a voltaje CD de polaridad conocida



Regulador de voltaje
Suaviza la salida del puente rectificador a 5 voltios de continua constante



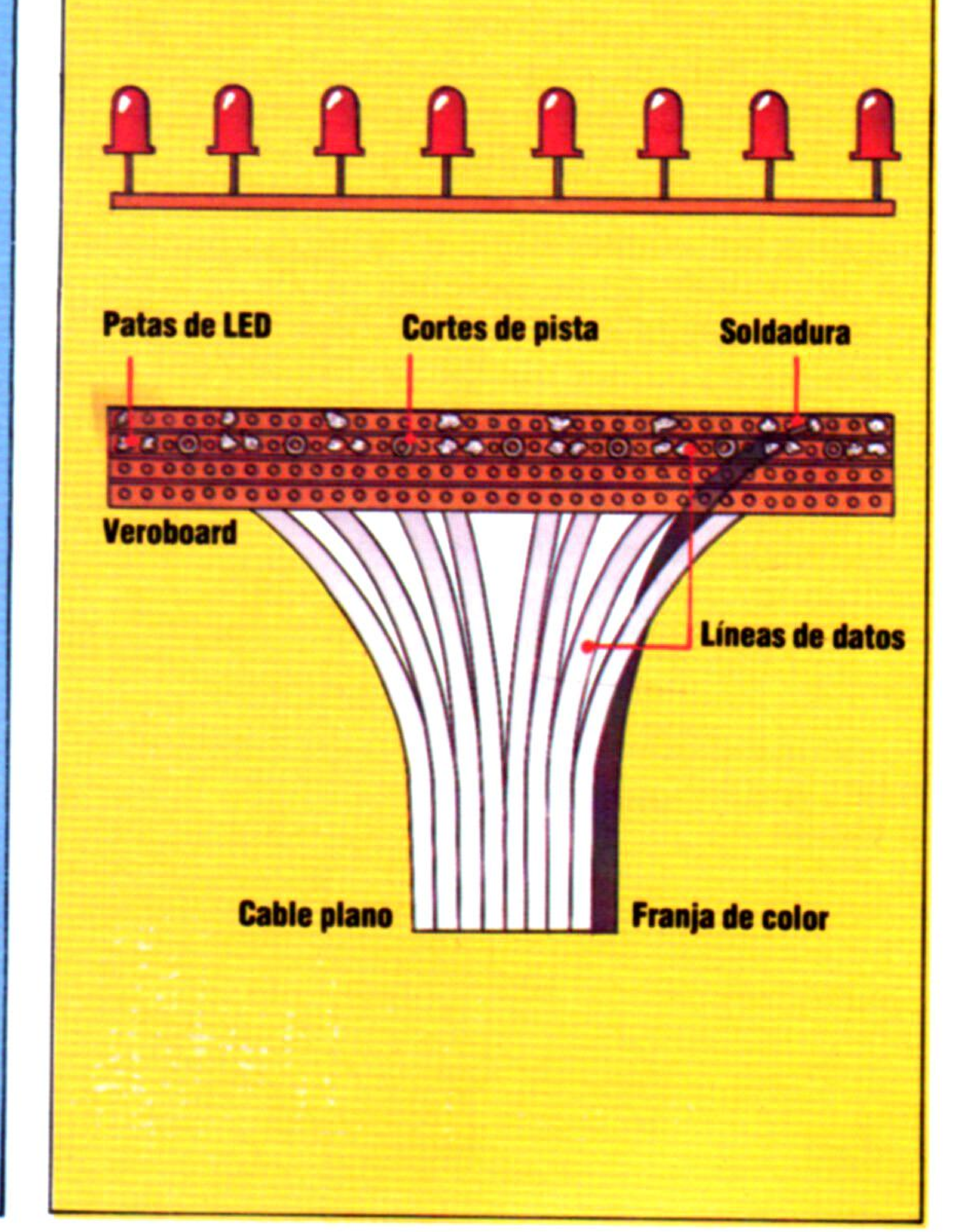
Conector de potencia
Acepta un enchufe de potencia de 2 mm, como el utilizado en las PSU (*Power Supply Unit*: fuente de alimentación) de muchos ordenadores, y se monta directamente en la placa de circuito impreso



Conector minicon
Se monta directamente en la placa y permite la fácil conexión de los cables externos

La visualización LED

Los LED ocuparán una franja de veroboard de cuatro pistas de ancho, teniendo cada pista 36 agujeros. Inserte los LED como muestra la ilustración, con las patas más largas en la pista del borde, dejando cuatro agujeros entre cada uno. La espaciación debe ser la misma que la de los agujeros taladrados en la caja; de ser necesario, vuelva a posicionar los LED. Suelde las patas a las pistas de cobre, procurando no arrastrar soldadura de una pista a la otra. Utilice un tester para comprobar la resistencia entre las dos pistas; si es cero, de alguna forma usted ha tendido un puente entre las pistas.
Corte un trozo de 20 cm del cable plano de 12 vías y quite tres alambres, dejando un cable de nueve vías incluyendo la de color. Pele y estañe los extremos de los cables. Suelde el cable de color a la pista del borde de la placa. Ahora suelde cada uno de los cables restantes a lo largo de la otra pista, cada cable junto a la pata de un LED. Corte esta pista en siete lugares, de modo que cada par de patilla y cable quede aislado en su pequeña franja de pista. Nuevamente, verifique si hay puentes entre pistas o entre las zonas separadas por cortes.
Pase suavemente los LED y la placa por los agujeros de la caja, atornille los conectores en sus agujeros, y después reclínese en la silla y admire su obra mientras espera el siguiente capítulo, en el cual le enseñaremos a construir la placa de circuito impreso





La caja del "buffer"

Conectores

Enchufes

LED

Placa de circuito impreso

Cable plano

Tapa de la caja

Conector de 12 vías

Tornillos de fijación

Conector

Rosalind Buckland

Líneas maestras

En esta ocasión analizaremos el diseño y la implementación de rutinas de "ayuda", que puede incorporar a sus programas

En nuestros días la memoria es barata. La próxima generación de ordenadores personales, que puede que tengan un *mínimo* de 128 Kbytes de RAM, nos permitirá disponer de mayor cantidad de memoria de la que jamás podría exigir ninguno de nuestros más ambiciosos programas. A través de la historia de la informática, la escasez de memoria ha sido la principal excusa que impedía proporcionar a los usuarios suficiente ayuda en forma de instrucciones, mensajes de error sensatos o ayuda en línea. En la actualidad ya no hay ninguna excusa.

Son tres las principales ayudas para el usuario que se pueden proporcionar en un programa: indicaciones de qué hacer, páginas de "ayuda" (*help*) y "postes indicadores". Las indicaciones o instrucciones asumen dos formas. Se pueden ofrecer en un único bloque al comienzo del programa, o se las puede ir suministrando a medida que se las requiera a lo largo del programa (como los avisos para una entrada del usuario, p. ej.). Lo ideal sería que el usuario pudiera tener ambas a su disposición.

En su forma más simple, las instrucciones podrían ser sencillamente una página (o varias) de texto explicando en español llano cómo utilizar el programa. El texto se puede almacenar en series o en sentencias DATA dentro del programa, visualizándose cuando así se solicite mediante una llamada a una subrutina escrita a tal fin. Al comienzo del programa se le pregunta al usuario si necesita instrucciones o ayuda; de ser así, se llama a la subrutina. De allí en adelante el resto de rutinas que acepten datos del usuario deben construirse de forma que con una entrada específica ("?" es común, o puede utilizarse "I") active una llamada a la subrutina de ayuda. Es una buena idea crear una instrucción estándar "visualizar ayuda", y modificar las rutinas de entrada de la biblioteca para aceptarla. No olvide modificar cualquier aviso que emplee en sus rutinas, de modo que "Pulse cualquier tecla para más..." se convierta en "Pulse cualquier tecla para más (o "I" para instrucciones)". Ello le facilitará un formato estándar para todos sus programas.

Pero no es necesario que las ayudas se compongan sólo de texto. Se pueden incluir diagramas y desarrollar las rutinas de ayudas o instrucciones como para que ofrezcan ejemplos sencillos y permitan que el usuario practique. Tales rutinas de ayuda son comunes en los programas que efectúan experimentos científicos; en este caso al usuario se le puede exigir que realice una tarea especificada con un determinado nivel de destreza antes de permitirle seguir con el programa principal. Estas rutinas de "enseñanza" no son fáciles de desarrollar porque deben simular el comportamiento del resto del programa, así como evaluar el rendimiento del usuario.



Palabras de consejo
El Micropro Wordstar ofrece un ejemplo muy vendido de software conducido mediante comandos con "ayuda en línea". El usuario puede abreviar el menú de ayuda en pantalla o bien prescindir directamente de él, pero mediante la sola pulsación de una tecla siempre se dispone de una estructura del archivo Help (de ayuda) sumamente detallada

Siguiendo un estilo similar, se podría llamar a las páginas de "ayuda" (*help*) para explicar la operatoria de determinadas partes del programa. Esta facilidad existe en muchos sistemas, que disponen de ella para explicar la utilización de las instrucciones; el sistema operativo Unix, por ejemplo, ¡permite acceder a todo el manual del usuario como ayuda en línea! Facilitar ayudas en sus propios programas no tiene por qué ser más difícil que proporcionar instrucciones: en cada punto apropiado, permita simplemente que el usuario entre una solicitud de ayuda en vez del input normal; cuando ello suceda, el programa habrá de llamar a la correspondiente rutina de ayuda. Es probable que un programa complicado requiera una gran cantidad de páginas de ayuda, de modo que, nuevamente, es deseable una rutina general de ayuda. Ésta podría pedir que el usuario entrase un número para identificar la página concreta de ayuda que se requiere. En los sis-

temas basados en disco, las páginas de ayuda se pueden almacenar como archivos separados. La rutina de ayuda creará entonces el nombre de archivo adecuado a partir de la entrada del usuario, leerá el archivo y lo visualizará en la pantalla.

Tanto las rutinas de ayuda como las de instrucciones bien pueden ocupar más de una simple página de información. De ser éste el caso, se debe diseñar la rutina de visualización de modo tal que el usuario pueda ir hojeando las páginas hacia atrás y hacia adelante a voluntad. También debe asegurarse que el usuario pueda abandonar la rutina y regresar al punto exacto en el cual salió del programa principal: ¡es muy frustrante tener que pasar por 10 páginas de información innecesaria cada vez que se necesitan instrucciones! Si se había dado un aviso, ahora se habrá perdido, y deberá repetirse. La rutina de ayuda debe establecer un indicador que le diga a la rutina llamada que debe volver atrás hasta la última instrucción previa a la llamada de ayuda, limpiando primero el indicador.



Ian McKinnell

Una metáfora común para las interacciones del usuario con programas complejos es la de pensar que éste va navegando a través de una enmarañada red de lógica. El recién llegado al programa no comprenderá su estructura y se puede desorientar y extraviar fácilmente. De modo que se necesitan "postes indicadores" para guiar al usuario. Un menú es el ejemplo más sencillo; opera como una señal vial que muestra las posibles salidas de un cruce. Sistemas como el Macintosh y el Lisa de Apple trabajan de forma similar, utilizando iconos en lugar de opciones de menús.

Algunas direcciones son más importantes que otras. En un sistema basado en instrucciones de ayuda puede haber docenas de posibles instrucciones. Sin embargo, no todas ellas serán relevantes o siquiera posibles en un determinado punto del programa. Si la cantidad de opciones es reducida, es útil visualizar una línea o dos para explicar qué son. Algunas opciones (como QUIT: salir) deben estar disponibles en todo momento, de modo que es una buena idea mantener éstas en pantalla. También pueden estar disponibles constantemente ciertas instrucciones específicas de las aplicaciones. Una técnica común consiste en programar tales instrucciones en teclas de función y visualizar un mensaje de una sola línea para señalar la función de cada tecla. Siempre es una buena práctica visualizar un poste indicador que señale la forma de salir de un programa: esto les da mayor confianza a los novatos, ¡para quienes la principal preocupación suele ser encontrar la salida de emergencia!



Buena gestión

El software Manager del ACT Apricot va guiando al usuario a través de un juego de hostiles programas de utilidades mediante su sistema de menús jerárquicos. La ayuda (Help) es una opción de todos los menús y consiste en una explicación de todas las otras opciones del menú. Éste es un buen ejemplo de software clásico guiado por menú apoyado por grandes archivos de ayuda

Se han desarrollado algunos sistemas experimentales que pueden comprobar el rendimiento de un usuario y ajustar el nivel de ayuda ofrecido en función de dicho rendimiento. Los programas comerciales que posean esta característica todavía están muy lejanos, pero es posible utilizar técnicas sencillas para alcanzar al menos una parte de este objetivo. Si se solicita al usuario que dé su nombre cada vez que se ejecuta el programa, entonces se puede llevar un archivo de usuarios y de sus niveles de destreza. Estos niveles se pueden calcular (a partir de la cantidad de veces que un determinado usuario ha ejecutado el programa o, por ejemplo, a partir de la máxima puntuación conseguida si el programa en cuestión es un juego) y actualizar al final de la ejecución del programa. A medida que aumente el nivel de destreza, se modificará el tipo de ayuda y la señalización con postes, haciéndose más concisos y menos intrusos. Al usuario también se le puede solicitar que elija el nivel de ayuda requerido, como en el paquete para tratamiento de textos Wordstar. En un plano ideal se utilizarían ambas alternativas.

La incorporación de ayuda puede ser una valiosa guía para mejorar el rendimiento de un programa. Una vez que se ha diseñado una rutina de ayuda es muy sencillo modificarla para que registre qué páginas de ayuda se emplearon y con qué frecuencia se las necesitó. Ello proporciona una clara indicación de los focos problemáticos del programa.

Respuestas adecuadas

Soluciones a los ejercicios propuestos en capítulo anterior (p. 996)

Todos los ejercicios de programación sugeridos se parecían al programa ejemplo "Números de teléfono" en cuanto a controlar la puerta para el usuario y comprobar su estado comparándolo con el de algún valor de referencia. Las soluciones que le presentamos no son, de ninguna manera, las únicas posibles

1) Alarma antirrobo

```
10 REM BBC VERSION 1.1
20 REGDAT=&FE60:RDD=&FE62
30 ?RDD=0:&REGDAT=127
40 REPEAT
50 PRINT "TODO VA BIEN"
60 ALARMA=?REGDAT
70 UNTIL ALARMA <> 255
80 PRINT "ALARMA"
90 FOR PIN=0 TO 7
100 IF (ALARMA AND 2^PIN)=0 THEN PRINT
    "ALARMA, INTERRUPTOR N."PIN
110 NEXT
120 END
```

```
10 REM CBM64 VERSION 1.1
20 RDD=56579:REGDAT=56577
30 POKE RDD,0:POKE REGDAT,255
40 FOR C=0 TO 1 STEP 0
50 PRINT "TODO VA BIEN"
60 ALARMA=PEEK (REGDAT)
70 IF ALARMA <> 255 THEN C=1
80 NEXT C
90 PRINT "SUENA LA ALARMA"
100 FOR PIN=0 TO 7
110 IF (ALARMA AND 2 ↑ PIN)=0 THEN PRINT
    "ALARMA, INTERRUPTOR N.";PIN
120 NEXT:END
READY.
```

2) Contador de impulsos (I)

```
10 REM BBC VERSION 1.2
20 REGDAT=&FE60: RDD=&FE62
30 ?RDD=0:?REGDAT=127
40 CONTADOR=0: REM INICIALIZAR CONTADOR
50 TIME=0: REM INICIALIZAR TIME
60 REPEAT
70 REPEAT
80 UNTIL ?REGDAT <> 255
90 CONTADOR=CONTADOR+1
100 UNTIL TIME > 6000: REM 1 MIN
110 PRINT "IMPULSOS="CONTADOR
120 END
```

```
10 REM CBM64 VERSION 1.2
20 RDD=56579:REGDAT=56577
30 POKE RDD,0:REM TODAS ENTRADA
40 T=TI: REM INIC. TIME
50 IF PEEK (REGDAT)=255 THEN 50
60 CONTADOR=CONTADOR+1
70 IF (TI-T) < 3600 THEN 50
80 PRINT "IMPULSOS=";CONTADOR
90 END
READY.
```

3) Contador de impulsos (II)

```
10 REM BBC VERSION 1.3
20 REGDAT=&FE60:RDD=&FE62
30 DIM C(8)
40 ?RDD=0:TIME=0
50 REPEAT
60 FOR N=0 TO 7
70 C(N)=C(N)+NOT(?REGDAT AND(2^N))
80 NEXT N
90 UNTIL TIME > 6000
100 FOR N=0 TO 7
110 PRINT "LINEA"N"="C(N)
120 NEXT N
130 END
```

```
10 REM CBM64 VERSION 1.3
20 RDD=56579:REGDAT=56577
30 POKE RDD,0:T=TI
40 FOR K=0 TO 7
50 C(K)=C(K)+NOT(PEEK(REGDAT)AND(2 ↑ K))
60 NEXT K
70 IF (TI-T) < 3600 THEN 40
80 FOR K=0 TO 7
90 PRINT "LINEA";K;"="C(K)
100 NEXT K
110 END
```

4) Cerradura con combinación

```
10 REM BBC VERSION 1.4
20 REGDAT=&FE60:RDD=&FE62:FLAG=1
30 ?RDD=15: REM LINEAS 0-3 ENTRADA
40 DIM C(3)
50 REM CODIGO=359
60 C(1)=3:C(2)=5:C(3)=9
70 FOR N=1 TO 3
80 A$=GET$:REM ESPERAR PULSACION TECLA
90 NEXT N
100 IF FLAG=1 THEN PRINT "ABIERTA" ELSE
    PRINT "COMBINACION EQUIVOCADA"
110 END
```

```
10 REM CBM64 VERSION 1.4
20 RDD=56579:REGDAT=56577
30 POKE RDD,15:REM LINEAS 0-3 ENTRADA
40 C(1)=1:C(2)=2:C(3)=4: REM CODIGO 124
50 FOR K=1 TO 3
55 PRINT"ENTRE UN DIGITO"
60 GET A$:IF A$="" THEN 60:REM ESPERAR PULSACION TECLA
65 PRINT A$
70 IF PEEK(REGDAT) <> C(K) THEN FL=1
80 NEXT K
90 IF FL=1 THEN PRINT "COMBINACION EQUIVOCADA":END
100 PRINT "ABIERTA"
110 END
```

5) Cambiar el color de la pantalla

```
10 REM BBC VERSION 1.5
20 MODE5
30 REGDAT=&FE60:RDD=&FE62
40 ?RDD=128: REM TODAS ENTRADA EXCEPTO D7
45 ?REGDAT=255
50 A$=GET$: REM ESPERAR PULSACION TECLA
60 COLP=?REGDAT:REM FONDOS > 127
70 GCOL0,COLP: REM CAMBIAR COLOR PANTALLA
80 CLG: REM LIMPIAR PANTALLA GRAFICOS
90 END
```

```
10 REM CBM64 VERSION 1.5
20 RDD=56579: REGDAT=56577
30 POKE RDD,0: REM TODAS ENTRADA
40 GET A$: IF A$="" THEN 40
50 COLP=PEEK(REGDAT)
60 POKE 53281,COLP
70 END
```



Una buena jugada

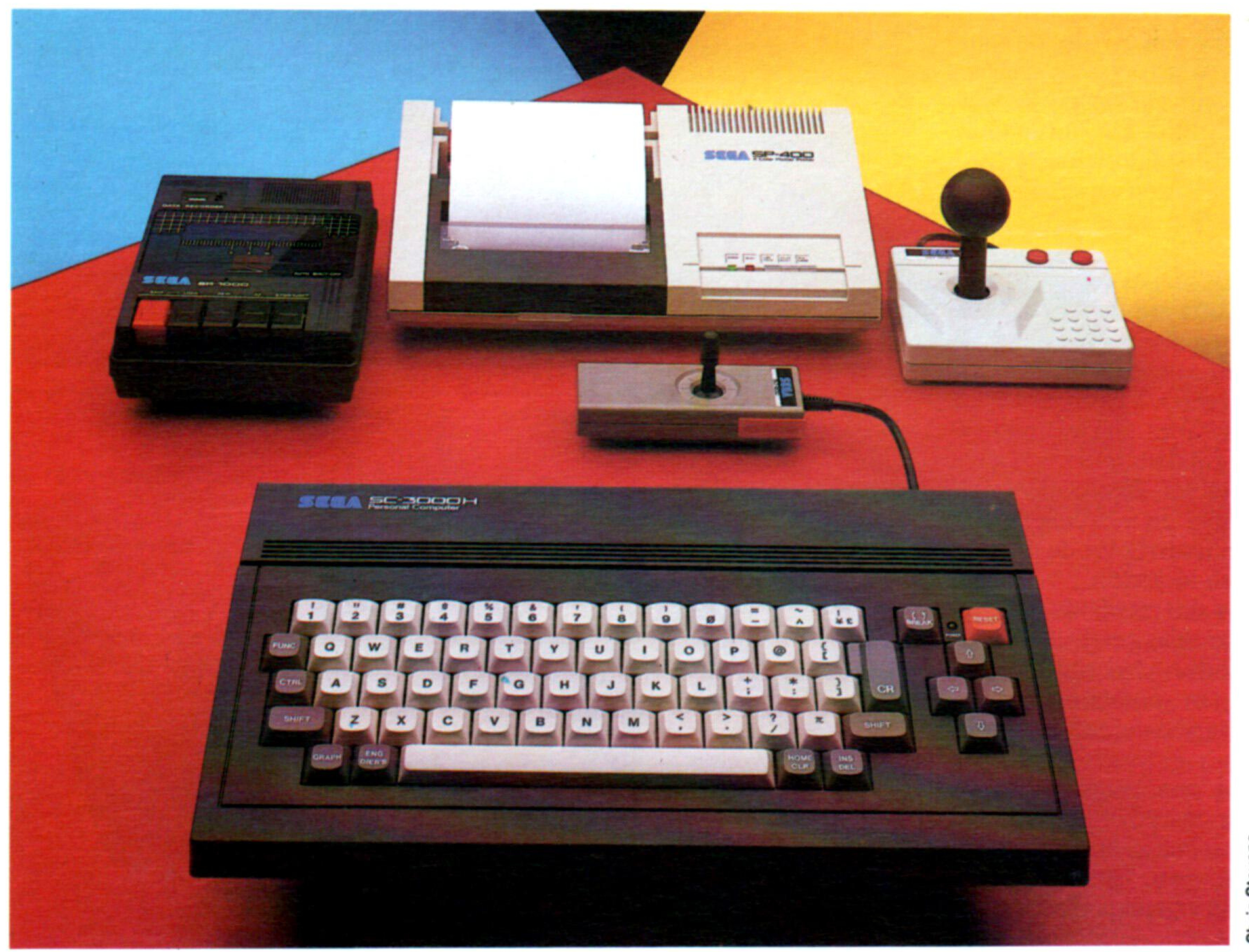
El Sega SC3000H es un micro de precio muy asequible, creado por una firma famosa por sus máquinas de juegos recreativos

El nuevo ordenador personal de Sega, el SC3000H, se ha presentado en diversas ferias de informática celebradas recientemente y ha sido objeto de comentarios favorables. Aunque es poco revolucionario en cuanto a diseño o función, utilizando el ya familiar procesador Z80A, se trata de un ordenador personal bien diseñado y fácilmente ampliable con gran cantidad de software disponible.

El Sega, una máquina atractiva y ligera que pesa 1,1 kg, tiene una carcasa plástica negra con teclas alfanuméricas blancas y teclas de operaciones grises (para funciones especiales, Control, Shift, Return, etc.). Posee teclas plásticas moldeadas tipo máquina de escribir, que al pulsarse recorren aproximadamente un centímetro. Las teclas hacen "clic" al ser pulsadas, lo que posiblemente sea un remanente del teclado de goma blanda utilizado en la versión japonesa. En líneas generales, no obstante, la calidad del teclado es buena para una máquina de su precio. Además de las teclas estándares, el Sega posee una tecla de función no programable que se utiliza para entrar palabras clave de BASIC, una tecla *graph* (de gráficos) para acceder a los símbolos para gráficos del teclado, teclas *clear screen* (limpiar pantalla) e *insert/delete* (insertar/eliminar) y un grupo de cuatro teclas para el cursor que resulta ideal para juegos. Sin embargo, un serio inconveniente es la ausencia de símbolos gráficos en las teclas. El SC3000 *soft-key* (de teclas blandas), que sólo se distribuye en Japón, tiene dichos símbolos impresos en las teclas; sin ellos la operación del SC3000H en BASIC resulta más dificultosa.

El SC3000H está bien equipado con interfaces. Contiene dos puertas para palanca de mando estilo Atari situadas sobre el lado izquierdo y una ranura para cartucho de ROM a la derecha, y las conexiones de la parte posterior de la máquina incluyen la salida para un aparato de televisión, una puerta para pantalla de color compuesto, otra para impresora tipo DIN, interface para cassette y conector de potencia para el adaptador de corriente de nueve voltios. También incluye una caja de conexiones para usar con un televisor y un cartucho de BASIC con un pequeño folleto de instrucciones.

La instalación del Sega es directa, lo que está muy a tono con la escasa documentación de ayuda. Un folleto de dos páginas describe la conexión del ordenador a un televisor y a la fuente de alimentación eléctrica. Un LED verde indica que la máquina está encendida, pero en el folleto no se mencio-



Recién llegado de Japón
Por su asequible precio se pretende que el Sega SC3000H entre en competencia con el Sinclair Spectrum y el Commodore 64. La máquina posee una gama completa de periféricos, incluyendo grabadora de cassette, palancas de mando e impresora-plotter en color

Chris Stevens

na que el Sega no funcionará a menos que se inserte en la puerta de ROM un cartucho que contenga BASIC o un juego. Además, hay intérprete interno de BASIC, y después de cargar el cartucho de BASIC el SC3000H queda con unos ridículos 515 bytes de RAM para el usuario: menos memoria que la que proporciona el ZX81 sin ampliar. Ésta es una clara desventaja en el mercado, porque significa que el usuario habrá de adquirir un módulo de ampliación si quiere utilizar la máquina para algo más que ejecutar cartuchos de juegos.

El manual podría asimismo inducir a confusión, dado que se refiere al teclado blando del SC3000 y es necesario que el usuario consulte el diagrama del teclado impreso para situar cualquier símbolo para gráficos. Este inconveniente se podría corregir en versiones ulteriores. El diagrama del teclado también muestra las palabras clave del BASIC impresas arriba o abajo de muchas de las teclas, al estilo del Spectrum. A ellas se accede manteniendo pulsada la tecla de función al mismo tiempo que la tecla alfanumérica que corresponda. De esta forma se puede acceder a instrucciones como RUN, LOAD y GOTO, a funciones matemáticas (ABS, SIN, COS, TAN) y funciones para series como LEFT$, RIGHT$ y MID$; pero, de nuevo, es necesario remitirse al manual constantemente.

En operación, los gráficos del SC3000H son impresionantes. La visualización configura dos pantallas: la pantalla de texto ofrece 24 filas de 40 columnas en dos colores, mientras que la pantalla para gráficos tiene una resolución de 256 × 192 pixels y es capaz de visualizar 16 colores. Los niveles de brillo se pueden ajustar para proporcionar hasta 210 matices y se pueden crear hasta 32 sprites. El BASIC Sega, que es similar al Microsoft Extended BASIC, utiliza las instrucciones DRAW, COLOR, PAINT y SPRITE para la programación de gráficos.

El SC3000H posee asimismo seis canales de sonido, que se pueden direccionar a través de instrucciones POKE o mediante la utilización de las instrucciones del BASIC BEEP y SOUND. Un cartucho de música sirve de ayuda para la composición, si bien las melodías producidas no exhiben en demasía la "calidad de sintetizador" a que hace mención el manual.

La experiencia de Sega en juegos recreativos se refleja en la calidad de su software de juegos. Los gráficos de éstos generalmente son buenos, aunque el SC3000H parece tener dificultad para visualizar textos y gráficos al mismo tiempo. Visualmente, los juegos guardan mucha semejanza con sus equivalentes recreativos, y la gran calidad del sonido hace que resulten aún más disfrutables. La tecla Reset se utiliza de una manera inusual: en vez de reiniciar un juego, funciona como un interruptor de palanca para introducir una pausa en la acción.

Sega suministra dos tipos diferentes de palanca de mando, pero ninguna de ellas proporciona un movimiento rápido y uniforme. No obstante, el grupo de teclas del cursor, tan sensatamente diseñado, facilita la sencilla operación del software para juegos desde el teclado.

La empresa comercializa una buena gama de equipos periféricos para la máquina, incluyendo una grabadora de cassette, impresora-plotter en color y una unidad de ampliación. Ésta proporciona 64 Kbytes extras de RAM y tiene una unidad de disco compacto incorporada.

Circuitería video

Entrada video
Se proporciona enchufe DIN para pantallas de video de color compuesto

CPU
El microprocesador Z80A está ubicado debajo de los circuitos de control de video

Modulador de TV
El SC3000H tiene un modulador RF incorporado para conexión a un aparato de TV normal

Puertas para palancas
Se proporcionan dos puertas para palancas de mando, compatibles con Atari, para el control de juegos

Chip de gráficos
Este chip confiere al SC3000H sus excelentes capacidades para gráficos y su generador de sprites

RAM
Aquí están situados los 8 K de RAM para el usuario y la RAM de video exclusiva del SC3000H



Palancas de mando del Sega SC3000H
Hay dos palancas de mando distintas para el Sega, cada una de ellas apropiada para diferentes estilos de juego. Ambas utilizan un mecanismo de interruptor de ocho posiciones para el movimiento del bastón y son compatibles con el conector estándar Atari de nueve patillas

Chris Stevens





Conector de impresora
Éste es un enchufe tipo DIN para conectar impresora-plotter en color de Sega y otras impresoras

E/S cassette
Si bien la mayor parte del software Sega está basado en cartucho, se proporciona control de cassette para que el usuario almacene programas

Adaptador de corriente

Puerta para cartuchos
La mayor parte del software para juegos de Sega se suministra en cartuchos de ROM que se conectan con el SC3000H a través de esta puerta

Control de E/S
Este chip controla muchas funciones de entrada y salida del SC3000H

ROM
Puesto que el Sega no posee un lenguaje BASIC incorporado, la ROM se utiliza fundamentalmente para operaciones internas y visualizaciones de pantalla

Safari Hunting (Cacería)
Cace feroces animales de la jungla con un fusil tranquilizador



Congo Bongo
Trepe por peligrosas pendientes persiguiendo al gorila. En este loco juego recreativo cuídese de los cocos que caen y de los molestos monos



Baseball
Batee contra el nutrido equipo contrario en esta realista versión





Sinbad Mystery (El misterio de Simbad)
Un juego que combina laberinto y aventuras exóticas

Software Sega
Los gráficos y el sonido del software suministrado para el SC3000H poseen la calidad que cabría esperar de una empresa de juegos electrónicos como Sega. Muchos de los juegos tienen excelentes visualizaciones tridimensionales y constituyen una innovadora reelaboración de los formatos de juegos estándar

SEGA SC3000H

DIMENSIONES
353×210×46 mm

CPU
Z80A, 4MHz

MEMORIA
8 K de RAM, ampliables internamente a 48 K;
18 K de ROM, ampliables internamente a 32 K;
16 K de RAM de video

PANTALLA
24 filas de 40 columnas de texto; resolución de gráficos de 256×192 pixels con sprites y 16 colores en 15 niveles de brillo

INTERFACES
Puerta para cartuchos de ROM, puertas para palancas de mando (2), TV y video compuesto, audio, cassette e impresora

LENGUAJES DISPONIBLES
BASIC, LOGO

TECLADO
64 teclas, tipo máquina de escribir, con juego de cursor, tecla de función, teclas para gráficos y caracteres especiales

DOCUMENTACION
Folleto de instalación de dos páginas y el manual que viene con el cartucho de BASIC. Este pequeño folleto es bastante descriptivo, pero no menciona que la máquina no hará nada hasta que se cargue un cartucho. Toda la documentación se refiere a la versión *softkey* del teclado

VENTAJAS
Se trata de una máquina agradable con notables gráficos y sonido y un BASIC bien equipado. Por su precio, es una buena máquina para juegos y un micro adecuado para principiantes

DESVENTAJAS
La documentación es insuficiente. Los 8 K de memoria hacen que programar sea virtualmente imposible. El teclado debería tener símbolos especiales impresos

El Sega SC3000H es una máquina de agradable uso y contiene algunas características útiles que no son habituales en un micro de su precio. Sega debería tomar medidas para mejorar la documentación, y el teclado necesita un cambio de estilo para que incluya etiquetas para palabras clave y gráficos. El principal inconveniente es, sin duda alguna, la limitada memoria para el usuario; en la actualidad la máquina sólo es apta para emplearla con el software para juegos de Sega: quien desee escribir programas necesitará ampliar la memoria interna o adquirir la unidad de ampliación.

Tortuga inquieta

Veamos cómo utilizar los gráficos tortuga del lenguaje LOGO para dibujar formas complejas con el mínimo esfuerzo

Abreviaturas

Muchas instrucciones de LOGO tienen abreviaturas; he aquí las correspondientes a las instrucciones que hemos mencionado:

FORWARD	FD
BACK	BK
RIGHT	RT
LEFT	LT
PENUP	PU
PENDOWN	PD
PRINT	PR
FULLSCREEN	FS
TEXTSCREEN	TS
SPLITSCREEN	SS

La primera versión del LOGO que salió para microordenadores fue el LOGO MIT; ahora se lo considera la versión estándar y lo produce Terrapin Inc. para las máquinas Apple y Commodore. Logo Computer Systems Inc. (LCSI) produjo otra versión para los ordenadores Apple, Atari y Spectrum, y pronto estará a la venta el LOGO LCSI para el BBC Micro. Existen otras versiones, pero estas dos son las más difundidas. Todos los programas que daremos a modo de ejemplo usan el LOGO MIT; cuando existan diferencias entre las versiones MIT y LCSI, las mismas se explicarán en el recuadro de "Complementos al LOGO".

Sólo hay una forma de aprender LOGO: ¡experimentando! Le sugeriremos algunas cosas para que vaya probando, pero lo mejor que se puede hacer es resolver problemas que uno mismo se haya planteado.

Después de cargado, el LOGO queda en modalidad "inmediata" y listo para recibir y obedecer órdenes (o comandos). En la mayoría de las versiones, estas instrucciones deben entrarse en letras mayúsculas. Digite DRAW y verá que la pantalla se divide en dos secciones (esto se denomina *modalidad de pantalla dividida*). La sección superior es para los gráficos; ocupa la mayor parte de la superficie de visualización, y en el centro se halla la "tortuga", representada mediante un pequeño triángulo. La sección inferior es para texto y de momento simplemente contendrá la indicación "?".

La tortuga es un objeto con el cual nos podemos comunicar dándole instrucciones. Si pensamos que es un "objeto" podremos comprender más fácilmente la programación con ella. Las cosas más importantes a considerar son la posición de la tortuga, hacia dónde apunta o encabezamiento (dirección) y si el "lápiz" que transporta está bajado (en cuyo caso dibujará una línea a medida que se mueva) o levantado (en cuyo caso se deslizará sin dejar ninguna huella). La digitación de DRAW posiciona la tortuga en el centro de la pantalla, mirando directamente hacia arriba y con el lápiz hacia abajo.

Ahora intentemos darle una instrucción:

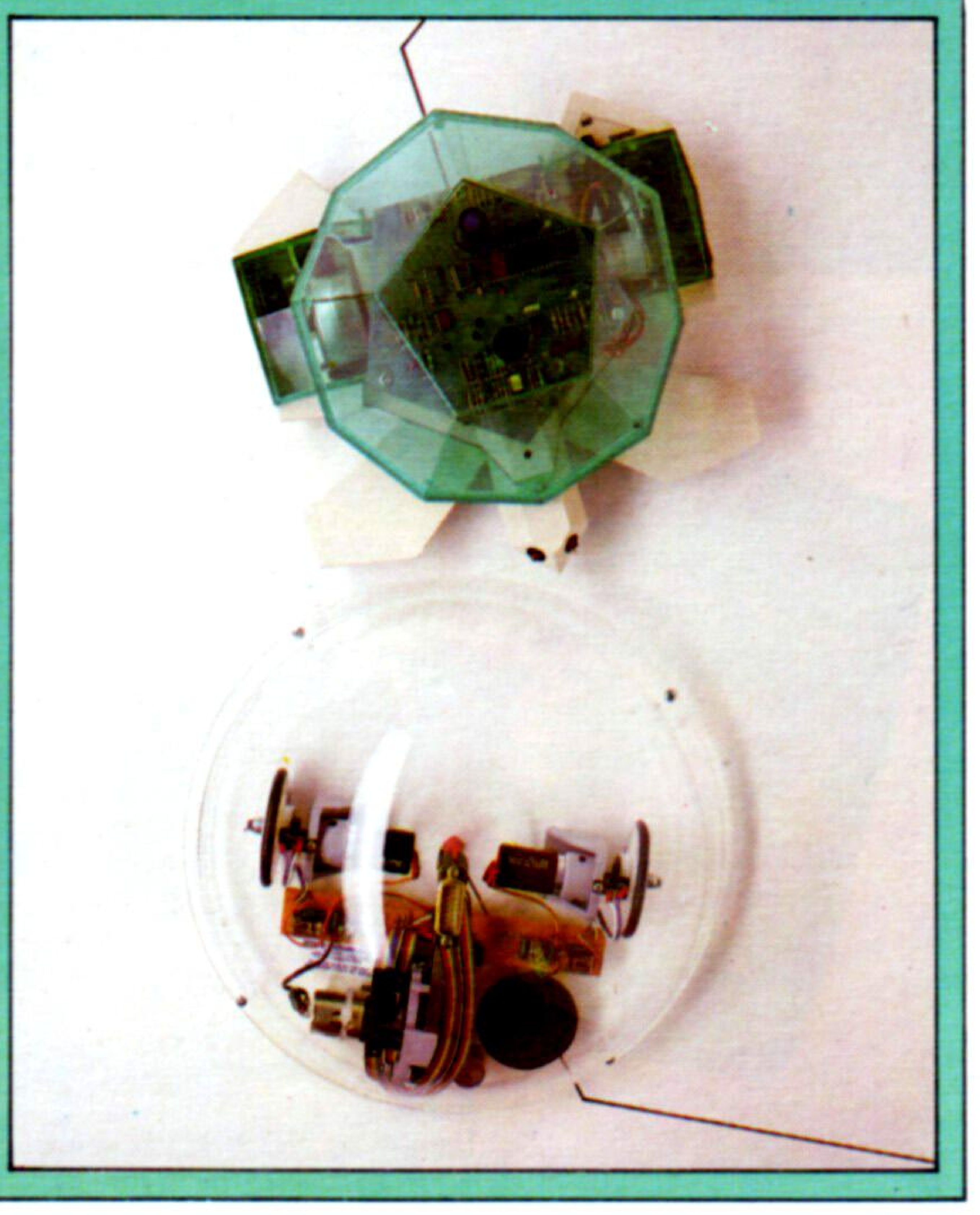
```
FORWARD 40
```

La tortuga se moverá 40 unidades hacia arriba de la pantalla, dibujando una línea a medida que avance. FORWARD es una instrucción de tortuga y el número 40 es su "entrada" (input). Unas órdenes necesitan entradas y otras no; DRAW, por ejemplo, no exige entrada.

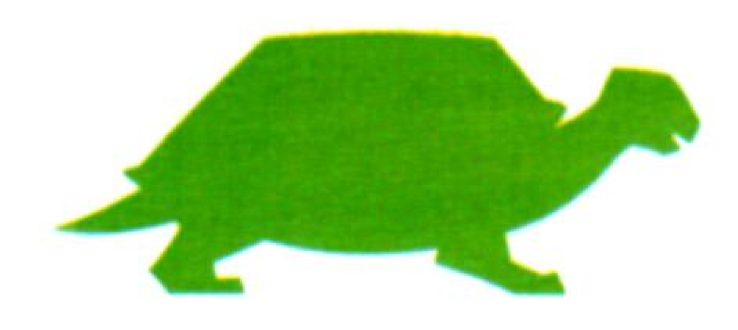
Una segunda instrucción de tortuga es BACK (atrás). BACK 10 le indica a la tortuga que se desplace 10 unidades hacia atrás. De modo que FORWARD y BACK (cada una con un número de unidades como entrada) modificarán la posición de la tortuga en la pantalla. RIGHT (derecha) y LEFT (izquierda), por otra parte, no modifican la posición de la tortuga, sino simplemente la hacen rotar; es decir, cambian la dirección en que está encarada. Estas dos instrucciones exigen como entrada un ángulo de entre 0 y 360°. Experimente un poco utilizando estas instrucciones; intente dibujar algunas formas simples; vea lo que sucede si le indica a la tortuga que se

Conozcamos la tortuga

Una tortuga es una herramienta robot de dibujo. Posee ruedas, está controlada por motores paso a paso y tiene un lápiz retráctil. Se la puede instruir para que se desplace hacia adelante, hacia atrás, a izquierda y derecha, y se puede elevar o hacer descender su lápiz. Cuando se baja, el lápiz produce un trazado. Cuando se desarrollaron por primera vez las tortugas tenían forma abombada, como la Edinburgh que vemos aquí, y se controlaban desde un teclado de ordenador. Esta tortuga se conecta al ordenador a través de un cable en paralelo. Los modelos más modernos son a control remoto. Ahora hay a la venta una versión de la tortuga Edinburgh controlada por radio. Y existe un robot con forma de tortuga, la Valiant Turtle, que posee una conexión de infrarrojos con el ordenador. Por extensión, la denominación de "tortuga" también se aplica al cursor de dibujo de la pantalla del ordenador en el LOGO. La mayoría de las tortugas de pantalla son simples formas triangulares, aunque el LOGO Atari visualiza un diminuto cursor en forma de tortuga



Paul Chave





desplace una distancia mayor de lo que permiten las dimensiones de la pantalla; intente utilizar como entradas números negativos. Para volver a empezar con la pantalla limpia, entre DRAW.

Cuando pruebe las instrucciones verá que la tortuga penetra en la sección de textos de la pantalla y, por tanto, parecerá hallarse por "detrás" del texto, de modo que no se la puede ver. Estas instrucciones le serán de ayuda:

FULLSCREEN (pantalla completa); permite usar la superficie total de pantalla para gráficos;
TEXTSCREEN (pantalla de texto): elimina todos los gráficos y deja sólo la modalidad de texto;
SPLITSCREEN (pantalla dividida): vuelve a la modalidad de pantalla dividida;
PENUP (lápiz levantado): permite que la tortuga se desplace sin dibujar ninguna línea;
PENDOWN (lápiz abajo): la tortuga va dejando una "huella" a medida que se desplaza.

Todas las instrucciones que hemos mencionado hasta ahora hacen que la tortuga obedezca sus órdenes. Pero también se puede utilizar el LOGO para obtener información de ella. La dirección a que apunta se mide en grados; un encabezamiento de 0 indica que la tortuga está mirando directamente hacia arriba, y el encabezamiento se mide en el sentido de las agujas del reloj a través de 360°. Para hallar el encabezamiento de la tortuga, digite:

PRINT HEADING (Imprimir encabezamiento)

La posición de la tortuga en la pantalla se define en términos de un sistema de coordenadas, con origen en el centro de la pantalla (es decir, comienza en un punto con las coordenadas *x* e *y* de 0). La posición se puede hallar en cualquier momento digitando:

PRINT XCOR PRINT YCOR (Imprimir coordenada *x* imprimir coordenada *y*)

Pruebe estas instrucciones dibujando una forma y cerciorándose luego de la posición de la tortuga y la dirección hacia la cual está orientada. Utilice estos datos para devolverla a su punto de comienzo.

Quizá se haya encontrado con que mientras experimentaba con instrucciones de LOGO, en la zona de textos aparecían mensajes de error. De no ser así, entonces cometa un error deliberado para ver lo que sucede. Por ejemplo, digite:

FORWARD50

Verá salir el mensaje:

THERE IS NO PROCEDURE NAMED FORWARD50
(No existe ningún procedimiento denominado FORWARD50)

La causa de esto es que el LOGO exige un espacio entre la instrucción FORWARD y la entrada 50, para eliminar cualquier confusión con una posible instrucción llamada FORWARD50. También puede obtener un mensaje de error si digita una instrucción en letras minúsculas.

El LOGO está equipado con un editor de líneas que permite que se corrijan instrucciones si se observa algún error antes de pulsar Return. Utilice las teclas del cursor para desplazarse a lo largo de la línea hasta el texto equivocado. Para insertar caracteres, simplemente digítelos: el texto situado a la derecha del error se desplazará automáticamente para hacer sitio a los caracteres extras. La tecla Delete elimina el carácter situado a la izquierda del cursor. Una vez corregida la línea, pulsando Return el LOGO aceptará la nueva instrucción. Si ya ha pulsado Return antes de detectar el error, digitando Control-P se recuperará la última línea para su edición. Esta característica es igualmente importante para la repetición de instrucciones.

Ahora podemos probar con algo un poco más matemático: por ejemplo, un cuadrado. Recuerde que un cuadrado tiene cuatro lados iguales y que sus esquinas son ángulos rectos (90°), de manera que algo como esto producirá el resultado deseado:

```
FD 50 RT 90
FD 50 RT 90
FD 50 RT 90
FD 50 RT 90
```

Observe que hemos abreviado las instrucciones y que podemos poner más de una en cada línea.

Lamentablemente, llegados a este punto puede que usted se encuentre con un problema técnico: que su cuadrado se parezca más bien a un rectángulo. Ello se debe a la "proporción de tamaños" *(aspect ratio)* de su pantalla; es decir, a la proporción del tamaño de paso vertical y el tamaño de paso horizontal. El LOGO posee una instrucción para manejar esto: utilice ASPECT seguida de un número (el valor por omisión es 0,8) para cambiar la proporción de tamaños hasta que su forma sea realmente un cuadrado. Ahora pruebe con los siguientes ejercicios: dibuje un triángulo equilátero, un pentágono, un hexágono, varios rectángulos, un rombo, un paralelogramo... en fin, ¡todo lo que se le ocurra!

Se podrían simplificar algunas de las instrucciones y reducir la cantidad de digitación mediante el empleo de REPEAT. Para volver a dibujar el cuadrado, digite simplemente:

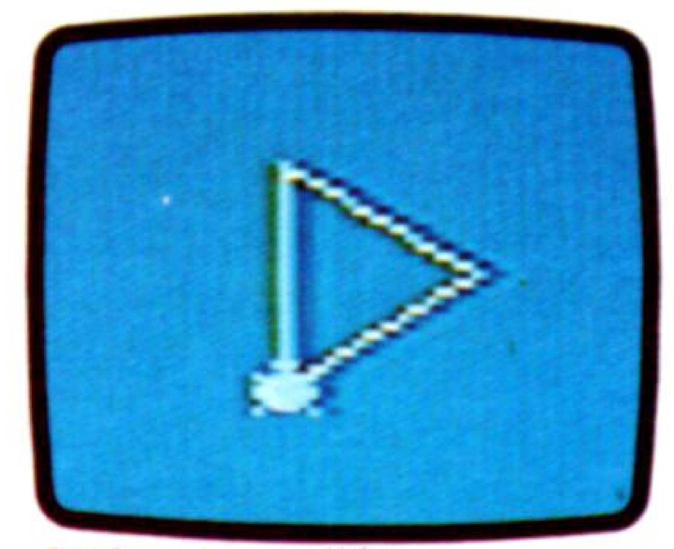
```
REPEAT 4 [FD 50 RT 90]
```

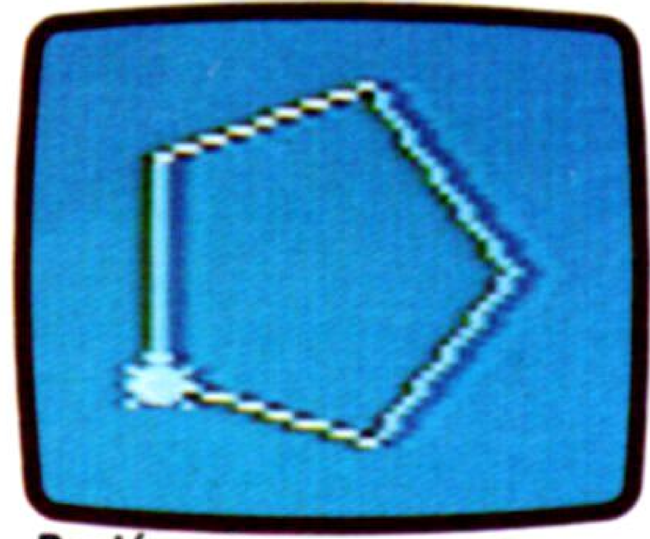
REPEAT (repetir) es una instrucción con dos entradas. La primera es un número que indica la cantidad de veces que el LOGO debe hacer algo, y la segunda es una "lista" de las instrucciones a obedecer. Esta lista siempre debe estar encerrada entre corchetes. De modo que nuestro ejemplo del cuadrado le dice al LOGO que debe repetir cuatro veces la secuencia FORWARD 50 RIGHT 90. Ahora intente simplificar la construcción de las formas que ya haya creado mediante la utilización de REPEAT, y vea si puede producir las formas de estrella que aparecen en algunas de nuestras ilustraciones.

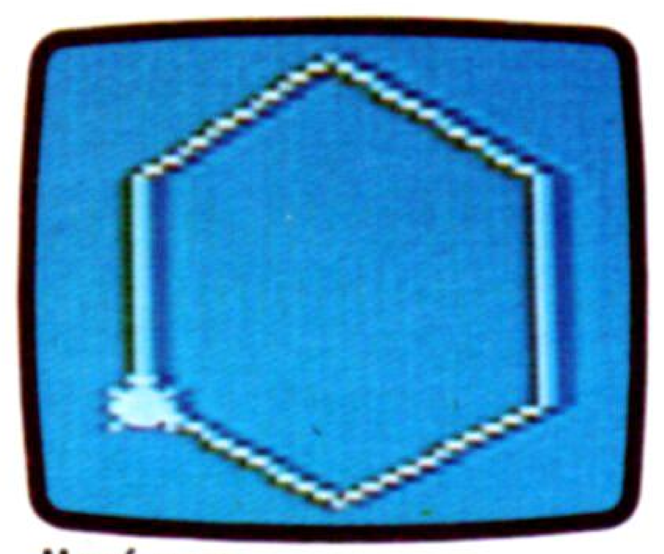
Complementos al LOGO

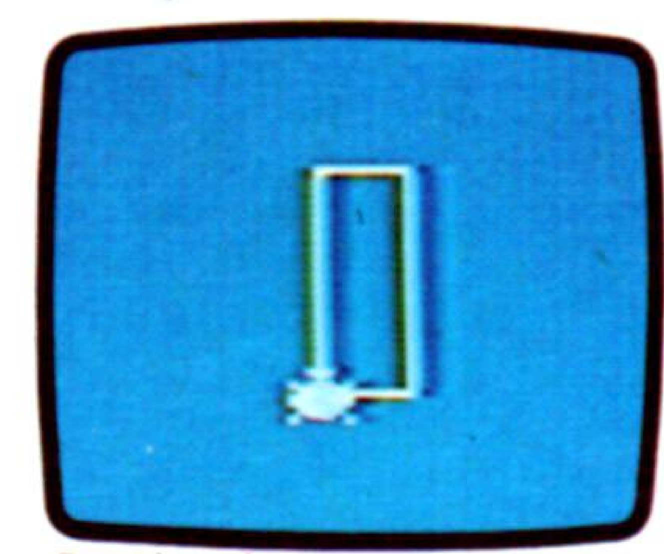
En el Atari, la tortuga tiene aspecto de tal.
En todas las versiones LCSI:
Utilice CLEARSCREEN (limpiar pantalla; abreviatura CS) para empezar a dibujar.
Utilice Control-Y para recuperar la última línea (excepto en el Spectrum, que no cuenta con esta posibilidad).
Para modificar la proporción de tamaños:
Apple LCSI — SETSCRUNCH seguido de la nueva proporción.
Atari —. SETSCR seguido de la nueva proporción.
Spectrum — SETSCRUNCH seguido de dos núms. de coords. [*x*,*y*] (la norma es [100 100])

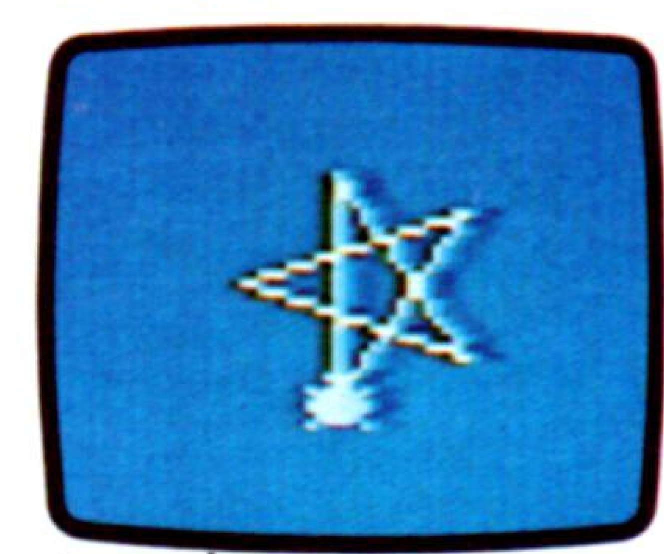
Ejercicios de LOGO
¿Puede escribir procedimientos para crear estas formas? Las muestras que ofrecemos se dibujaron con LOGO LCSI en un Atari 600 XL


Triángulo equilátero

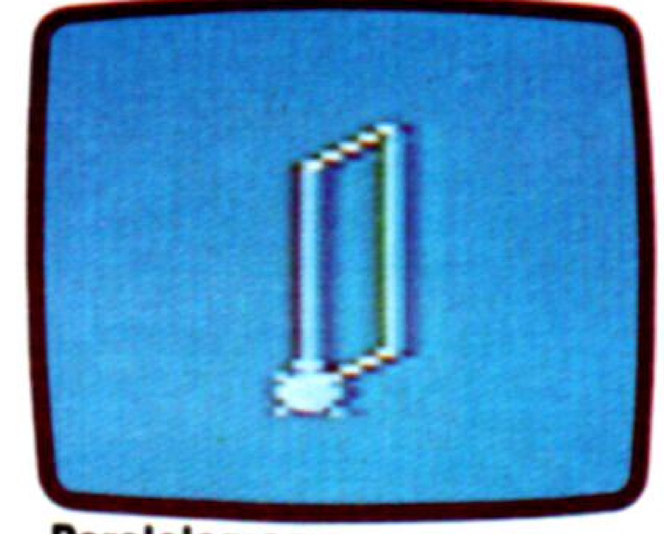

Pentágono


Hexágono


Rectángulo


Estrella de cinco puntas


Estrella de diez puntas


Paralelogramo

Ian McKinnell

Sonidos en secuencia

En este capítulo analizaremos un desarrollo de la secuenciación: la interface MIDI

El secuenciador ha tenido un impacto dramático en la composición musical, tanto en las actuaciones en vivo como en los estudios de grabación. Pero el inconveniente del secuenciador es que puede controlar una gama de variables en un solo sintetizador. Si un músico de teclado posee dos sintetizadores, uno con poderosas capacidades de secuenciación y el otro con buen sonido, no existe forma alguna de conectarlos en una única unidad coordinada. El problema es más grave en los estudios que necesitan coordinar diversos sonidos provenientes de distintas piezas del equipo. La finalidad de una interface digital para instrumentos musicales es la de realizar esta conexión y colocar el control del sistema en manos de una sola máquina de enorme capacidad. La MIDI constituye un intento de conseguir esto en un formato estandarizado, de manera que cualquier sistema digital pueda controlar a otro en la producción de música.

La MIDI (*Musical Instrument Digital Interface*: interface digital para instrumentos musicales) se desarrolló después de una serie de reuniones que celebraron los principales fabricantes japoneses y norteamericanos de instrumentos digitales. Las especificaciones actuales fueron terminadas en agosto de 1983. Está diseñada como un circuito de control para transmitir datos de un instrumento a otro, o desde un microordenador a un instrumento.

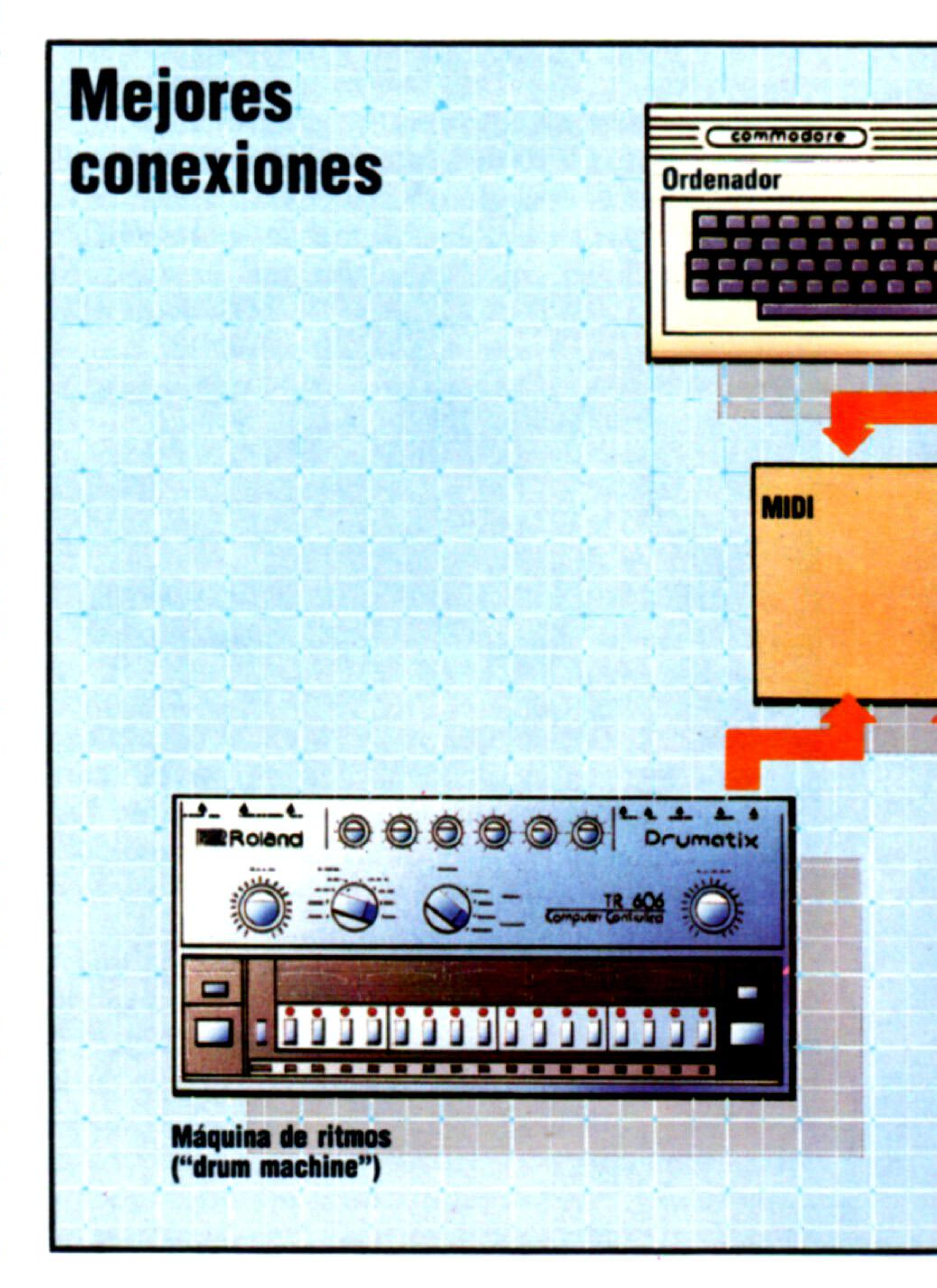
La MIDI opera con un procesador de ocho bits. Sus diseñadores tuvieron que elegir entre los dos medios de transmisión disponibles: en paralelo o en serie. La transmisión en parelelo proporciona líneas individuales, de modo que los ocho bits de cada byte de datos se envían simultáneamente y, por consiguiente, permite una gran velocidad de transmisión. Su desventaja reside en el costo extra que implica y el inconveniente de requerir al menos ocho líneas cableadas en un conector tipo D de 25 patillas. La transmisión en serie sólo utiliza dos líneas. En una los bits de datos se envían uno después del otro; se proporciona una segunda línea para que el instrumento receptor avise los errores de paridad (véase p. 667) en la corriente de datos que vuelve al instrumento maestro o microordenador. Así, la transmisión en serie es más lenta que la transmisión en paralelo, pero tiene la ventaja de un conector más sencillo y de ser mucho más barata.

La razón de ser original de la MIDI fue proporcionar un circuito de control que pudiera realzar la música sintetizada y proporcionar alguna forma de control de altura (*pitch*). Su índice de comercialización había de estar situado en una gama de precios asequible a la mayoría de los usuarios de ordenadores personales y de los músicos que utilizaran sintetizadores y máquinas de ritmos. Fundamentalmente fue éste el motivo por el cual se adoptó para la MIDI la transmisión en serie.

La MIDI transmite de forma asíncrona, a lo largo de las mismas líneas que la interface RS232, que es estándar para conectar muchos microordenadores con modems o impresoras en serie. Cuando se transmiten datos de forma asíncrona, ha de definirse cada byte para el instrumento receptor. En la MIDI esto lo efectúa un chip ACIA (*Asynchronous Communications Interface Adaptor*: adaptador para interface de comunicaciones asíncronas) Motorola 6850, que agrega dos bits extras a cada byte del instrumento maestro. Empieza en "0", el bit de comienzo (*start*), seguido de los ocho bits de datos en serie, y termina en "1", el bit de final (*stop*). Esta palabra en serie de 10 bits se transmite entonces al instrumento receptor, donde un segundo chip ACIA lo vuelve a convertir en los ocho bits de datos reales.

El costoso chip ACIA está protegido dentro del circuito mediante optoaislamiento. Un *optoaislador* en un dispositivo que utiliza células fotoeléctricas para permitir que dos circuitos no conectados intercambien señales permaneciendo, no obstante, eléctricamente aislados; las "ondas" de voltaje no dañan, en consecuencia, al chip.

La MIDI difiere del diseño de la interface RS232 en un importante sentido. La velocidad de transmi-







Glorioso Gliss
El *glissando* (deslizar el sonido de una nota a otra) se realiza fácilmente en un instrumento de cuerdas dejando resbalar los dedos hacia arriba o hacia abajo del traste. Para producir esto como un efecto de "resbalamiento", y no como una serie de notas marcadas, un sintetizador necesita una cuidadosa programación

sión de datos con la RS232 es de 1 920 palabras en serie por segundo, o 19,2 Kbaudios; la MIDI es un 50 % más lenta. Ello no afecta la compatibilidad con ordenadores personales, porque el sistema de circuitos lógicos dentro de la propia MIDI establece una nueva velocidad de reloj de 3 125 palabras por segundo, o 31,25 Kbaudios. Esta velocidad es relativamente elevada para la transmisión en serie, pero no es lo suficientemente rápida. Más adelante, en este mismo capítulo, veremos por qué.



En un sistema de música, el ordenador puede jugar dos posibles papeles, según la unidad MIDI en uso: en primer lugar, antes que nada debe colocar a la MIDI en modalidad activa y proporcionar la base de memoria para grabar la música; en segundo lugar, ha de soportar el software MIDI si el mismo no está residente en la propia unidad. Del mismo modo, la MIDI puede ser simplemente una interface digital entre los instrumentos musicales y el ordenador, o puede ser una interface con control activo sobre los datos transmitidos.
Existen dos modalidades de transmisión: grabación (*record*) y reproducción (*playback*). En la primera, la música producida con los instrumentos se envía a través de la MIDI a la memoria (ya sea del ordenador o del buffer de la MIDI). En la segunda, esta información digital es procesada por la MIDI de camino hacia los instrumentos: la información de tiempo, sincronización y control se le acopla a la misma tal como fue especificada en el protocolo definido por el usuario

Kevin Jones

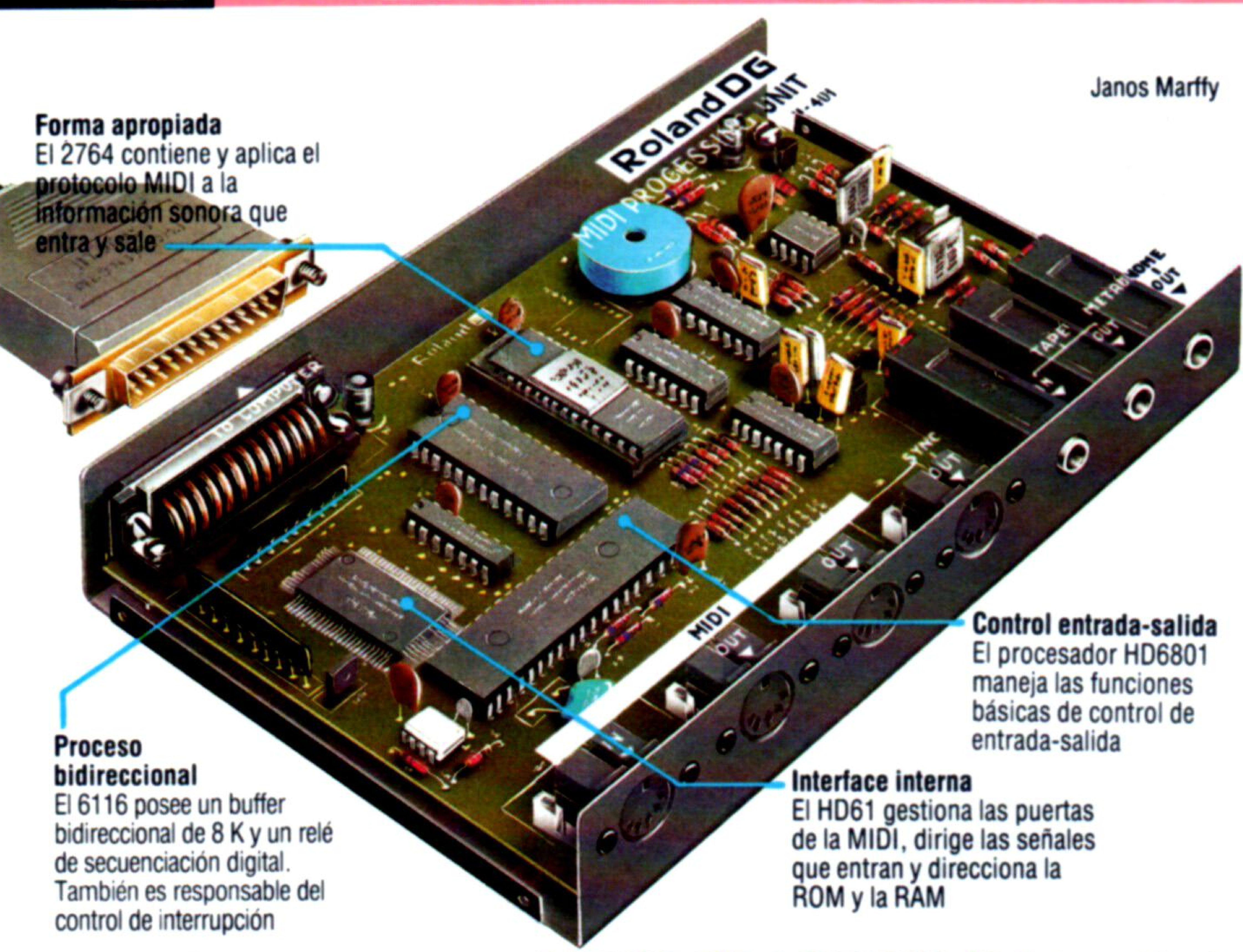
La MIDI está diseñada para la conexión en interface con más de un instrumento receptor. Cuando hay más de un instrumento recibiendo instrucciones de la MIDI, la primera exigencia debe ser que cada uno de los datos se envíe al instrumento apropiado; de lo contrario, una máquina de ritmos podría acabar intentando tocar una melodía cuidadosamente secuenciada, y un sintetizador polifónico reproduciendo un patrón de bombo en *do central*. Los instrumentos compatibles con la MIDI deben poseer un ID o código de identificación numérico. Al código se le asigna uno de los 16 canales disponibles de la MIDI, de modo que solamente ese canal acepte datos para ese instrumento. La primera parte de una transmisión MIDI completa es, entonces, un byte de estado que incluye el ID. Todos los datos que siguen a esta instrucción de ruta pueden entonces especificar cómo se debe interpretar la instrucción.

La unidad, con un tamaño aproximado de 100×120×45 mm, posee dos conectores DIN de cinco patillas, señalados como "MIDI IN" y "MIDI OUT". El primero acepta todas las instrucciones provenientes de un microordenador o sintetizador maestro, y "MIDI OUT" transmite al instrumento receptor la corriente de bits modificada. Muchos modelos poseen, asimismo, "MIDI THRU", un segundo conector de salida que simplemente transmite la corriente de bits original sin modificar enviada a "MIDI IN". Ésta se puede enviar entonces a una segunda interface. El cable, de una longitud máxima de 15 metros, está equipado con enchufes DIN de cinco patillas, y se conecta en el panel posterior de un microordenador o sintetizador maestro.

Do central

Pensemos en alguien que utiliza por primera vez una MIDI y desea probar una breve melodía. Ésta comienza en *do central*, sube a *mi*, después a *sol*, etc. La forma de indicar esto dependerá del tipo de software de música que esté utilizando. Tal vez esté empleando un lápiz óptico para ir marcando las notas en un pentagrama visualizado en la VDU. Este pentagrama se ha venido utilizando como formato de notación estándar en la música occidental desde hace más de cuatro siglos. Quizá esté entrando información en el teclado alfanumérico de su microordenador, empleando alguna clase de MCL (*Music Composition Language*: lenguaje para composición musical), nuevamente con una visualización VDU. Otra alternativa sería que estuviera tocando las notas de la melodía en un periférico de teclado musical. Este teclado podría no tener sonido propio, sino que produciría una u otra de las visualizaciones mencionadas anteriormente. Pero, cualquiera que sea la forma de entrar la música, la transmisión MIDI siempre será la misma: para empezar la melodía, el primer byte (transmitido como una palabra en serie con sus dos bits extras del chip ACIA) generará PLAY / ON CHANNEL 6; el segundo byte, MIDDLE C (tocar/en canal 6, *do central*, o *do*$_3$).

Esta mínima instrucción producirá la nota *do central* del instrumento receptor. Y el sintetizador continuará tocando *do central* a menos que también haya una instrucción que limite su duración, como STOP PLAYING / ON CHANNEL 6 (dejar de tocar en canal 6), byte uno; MIDDLE C *(do central)*, byte dos;



Janos Marffy

Forma apropiada
El 2764 contiene y aplica el protocolo MIDI a la información sonora que entra y sale

Control entrada-salida
El procesador HD6801 maneja las funciones básicas de control de entrada-salida

Proceso bidireccional
El 6116 posee un buffer bidireccional de 8 K y un relé de secuenciación digital. También es responsable del control de interrupción

Interface interna
El HD61 gestiona las puertas de la MIDI, dirige las señales que entran y direcciona la ROM y la RAM

La interface digital para instrumentos musicales
La interface MIDI concilia los protocolos de entrada-salida del ordenador y de los instrumentos musicales a él conectados (exactamente igual que cualquier otra interface), posibilitando, por consiguiente, que los instrumentos utilicen la memoria del ordenador. También procesa el sonido digitalizado que pasa a través de ella, agregando información relativa a control, sincronización y tiempo a la entrada del sintetizador

y ALLOWING FOR A DURATION OF X (permitiendo una duración de X), byte tres. Si no se suministra esta instrucción, y el resto de la melodía, *mi*, *sol*, etc., se entra sin parámetros de duración, todas las notas seguirán sonando. El resultado será un acorde mantenido compuesto por las notas entradas.

Afortunadamente, incluso un somero conocimiento de la notación de pentagrama en la VDU será suficiente para indicar que lo que se pretendía que fuera una melodía es probable que se convierta en un acorde. Y un usuario de MCL que haya incurrido en tal error podrá simplemente ejecutar la secuencia, pero esta vez enviando una instrucción MONOPHONIC (monofónico) al sintetizador. *Monofonía* significa sencillamente un sonido, por oposición a muchos (*polifonía*). Si el sintetizador receptor puede producir sólo un sonido por vez cuando se establece en MONOPHONIC, la secuencia, pobremente introducida, se puede ejecutar como una sucesión de notas individuales solamente; una melodía, más que un acorde.

Supongamos que, al cabo de algunos ensayos y errores, el usuario novel ha entrado su frase musical correctamente, y que el sintetizador en interface esté ahora tocando. La melodía conserva el compás, y el ritmo (definido por la secuencia de duraciones) es correcto. Hay que destacar que hasta el momento de nuestro análisis los tipos de instrucción han sido bastante limitados. Sólo hemos hecho alusión a dos parámetros o características musicales: altura (*do central*, *mi*, *sol*, etc.) y duración.

El compositor de la melodía la escucha varias veces, y le parece que suena un poco "dura" (como es probable que suene mientras sólo posea un mínimo de definición). Opta, entonces, no por que del primer *do* se pase al *mi* secamente, sino por que desde el *do* vaya subiendo el tono gradualmente hasta el *mi*. Esta clase de movimiento se denomina *glissando* o deslizamiento de tono, y será característico de la forma en que una persona entone la melodía. En este contexto se dará a la actuación del sintetizador un toque de agilidad. De modo que ahora esta instrucción sustituye a la original para *do central*, agregando un byte extra.

Esto nos lleva a un punto muy simple que concierne a la interface MIDI. Si el sintetizador receptor no posee facilidad para producir *glissandi*, no puede ejecutar esta última instrucción. Podría tocar el *do central* tal como si estuviera recibiendo la instrucción *original*, o bien hacer algo completamente diferente. Si las instrucciones del usuario de una MIDI han de producir una sección de música polifónica y el sintetizador receptor es un instrumento sólo monofónico, es probable que éste haga una selección impredecible de la polifonía y luego la toque de forma monofónica. En resumidas cuentas, utilizar la MIDI para conectar un microordenador con un sintetizador muy básico no convertirá a éste en un sintetizador caro como el Fairlight.

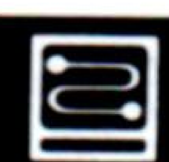
Estas limitaciones también se aplican en sentido inverso. Puede que el instrumento receptor sea un soberbio y caro sintetizador, pero a menos que se definan suficientes parámetros musicales y que se establezcan de la forma deseada los propios controles del sintetizador, el resultado bien podría sonar con la musicalidad de una calculadora de bolsillo.

En la práctica, la segunda de estas dos situaciones es muy fácil de mejorar. Se deben establecer como *constantes* la mayor cantidad posible de parámetros utilizando los controles del sintetizador, y las instrucciones de la MIDI deben trabajar dentro con esos parámetros. Lo más probable es que sea éste el enfoque que adopte el músico de sintetizador cuyos problemas hemos considerado antes.

Hasta este momento hemos analizado las características de altura y duración, pero la MIDI puede estar provista hasta de 128 controles teóricos, que cubren filtración, distorsión, "ruido blanco" (todas las frecuencias posibles) y "ruido rosa" (frecuencias de escala media), cada uno de ellos con valores comprendidos entre 0 y 128. Esto es más que adecuado para tratar con los parámetros de que disponen la mayor parte de los sintetizadores, y son estos controles los que probablemente les interesen a los usuarios de microordenadores.

Y es aquí donde la velocidad de transmisión de la MIDI adquiere relevancia. Hemos visto que una instrucción muy directa, relativa a una única nota y definiendo sólo dos parámetros, utilizaba tres palabras en serie. Con la velocidad de 31,25 Kbaudios, esto lleva casi un milisegundo. Los acordes de seis notas son comunes en muchos tipos de música: transmitir tal acorde ocuparía 5,76 milisegundos. Si ahora comenzamos a definir aún más este acorde utilizando controles MIDI, el tiempo de transmisión se vuelve demasiado lento y el oído humano comienza a detectar cambios en las características del sonido, producidos por la demora. Estos cambios se hacen evidentes sólo cuando los sonidos, en especial los similares, se producen juntos; pero como interface de audio, la MIDI se diseñó para manipular sonidos simultáneos. La música, lamentablemente, es un medio "en paralelo": como oyentes, estamos habituados a oír que las cosas sucedan de forma simultánea.

Por consiguiente, no es sorprendente que la transmisión en serie de la MIDI haya sido objeto de críticas; la transmisión en paralelo habría realizado mejor la tarea. Queda por ver si a los usuarios de la MIDI este fallo les molesta. En el presente, el diseño de la interface parece mantener un "delicado equilibrio" entre costo y eficacia, de modo que vale la pena tener presente que las especificaciones actuales bien podrían ser sólo las primeras.



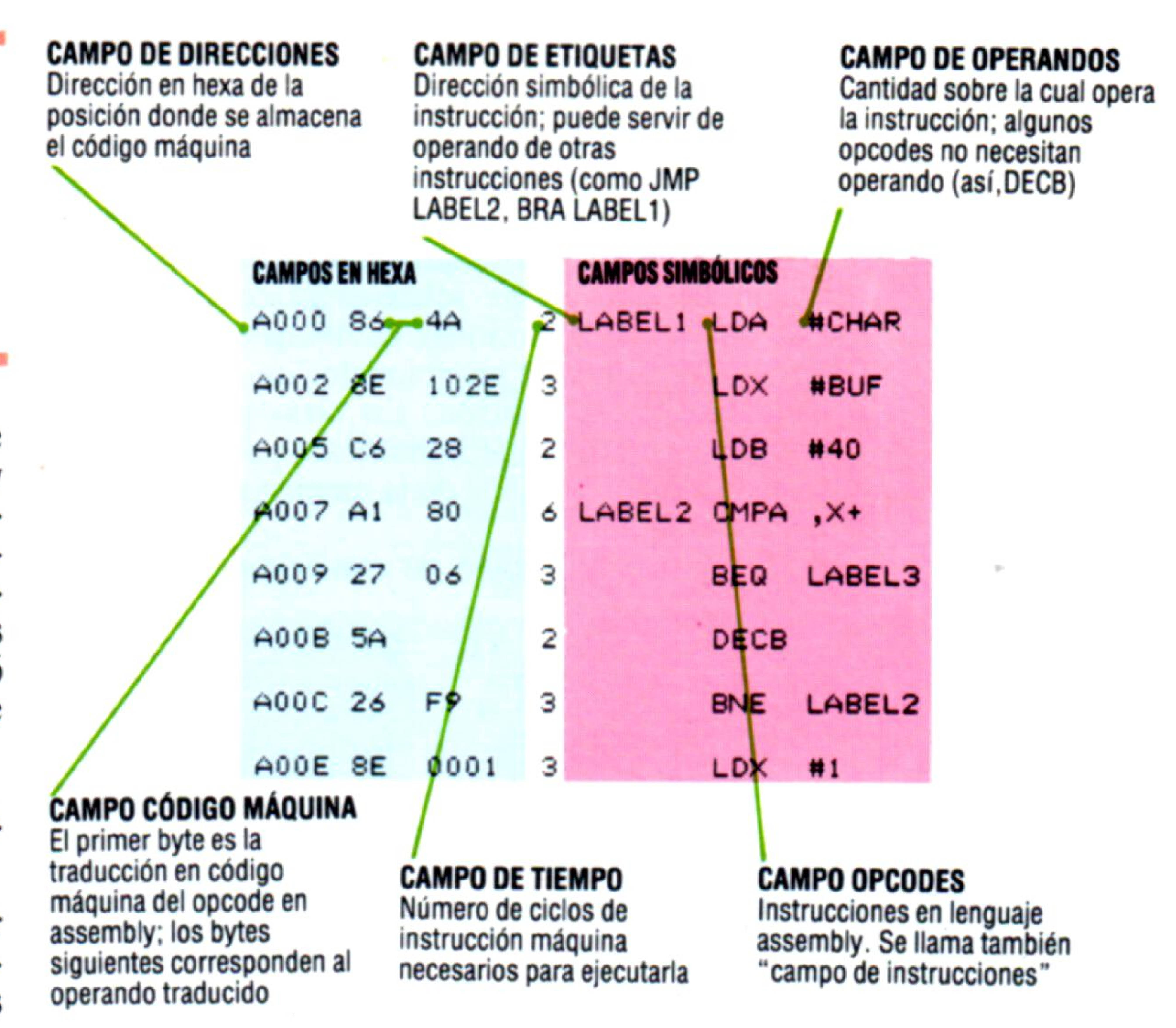
Cuestión de registros

Vamos a estudiar los registros del 6809 con más detalle, analizando su funcionamiento y algunas de las instrucciones básicas que suelen utilizar

Ya hemos visto que un registro es una posición de memoria dentro del mismo chip del procesador, y observamos cómo la programación en assembly implica el manejo de valores almacenados en los registros y su traslado de o a la memoria. Vimos posteriormente que algunos registros (en especial los que almacenan y procesan direcciones) son de 16 bits, mientras que otros son de ocho bits, y que cada registro realiza una función distinta.

- Los **registros índice** sirven para modificar las direcciones empleadas en nuestro programa.
- Los **índices de pila** son usados por el procesador como zona de trabajo y el programador los puede utilizar para recuperaciones y almacenamientos rápidos.
- El **contador del programa** retiene la dirección de la instrucción siguiente a ejecutar, pudiéndose alterar el flujo secuencial del programa modificando su contenido.
- Los **acumuladores** son los registros más frecuentemente utilizados, y se emplean para realizar funciones aritméticas.
- El **registro de código de condición** contiene varios flags que representan el estado del procesador (por ej., si es cero el resultado de una operación). Estos flags pueden ser consultados con el fin de hacer una selección o un bucle, proporcionando al assembly una estructura IF...THEN.

El procesador 6809 contiene todos estos registros. Puesto que se trata de una variante mejorada del primitivo procesador Motorola 6800 (al igual que el 6502, empleado por el BBC Micro y muchos otros), existen muchas similitudes entre los lenguajes assembly de ambos procesadores. Sin embargo no son compatibles y la codificación escrita para el uno no podrá ser ejecutada por el otro. Muchos programas del 6800 tienen el código fuente compatible: un programa en assembly escrito para el 6800 puede ser reensamblado para el 6809 con buenas posibilidades de que funcione. Pero ni siquiera este pequeño peldaño de compatibilidades está al alcance del 6502 (o de su posterior desarrollo, el 6510 que emplea el Commodore 64). De todos modos, las similitudes entre ambos procesadores hacen al menos que la tarea traductora de un programa en versión del 6809 no sea demasiado difícil, y puede servir de buena introducción al 6809 para el que ya esté familiarizado con el 6502. Los registros que contiene el 6809 son:



Campos de estudio
Los programas en lenguaje assembly se parecen muy poco a los listados de programas en BASIC, y una vez traducidos a código máquina mediante un programa ensamblador todavía aparecen más desconcertantes. La clave para su comprensión es fijarse en las columnas (o campos) adecuadas (por lo general, los campos de etiquetas, de operandos y de opcodes), ignorando el resto. Esta muestra le puede ayudar

- Dos acumuladores de ocho bits, conocidos como A y B. No existe diferencia funcional alguna entre ellos, pudiéndose usar cualquiera de los dos, cuando son empleados como registros de ocho bits. Pero el hecho de que sean dos permite retener valores en uno de ellos mientras se trabaja con el otro. O también pueden usarse conjuntamente como si fuera un único acumulador de 16 bits, facilidad poderosa que permite realizar operaciones aritméticas directamente en 16 bits. Precisamente por este motivo se dice que el 6809 es un pseudoprocesador de 16 bits, aunque esto no es así, sino más bien un avance dentro de la línea de los procesadores de ocho bits. Si se emplean juntos A y B, son llamados registro D de 16 bits.
- Dos registros índice, el X y el Y. De nuevo nos encontramos con que su funcionamiento es idéntico, siendo indiferente el empleo de uno u otro. Sin embargo, existe una ligera diferencia operativa, ya que algunas instrucciones que se sirven de Y se traducirán como instrucciones de dos bytes, mientras que las correspondientes al registro índice X serían de un solo byte, haciendo de esta manera el programa un poco más largo y lento. Siempre que se necesite solamente un registro, es preferible utilizar X.
- Dos índices de pila, S y U. El procesador emplea S para todas sus operaciones con la pila. Por eso, el programador, aunque puede usar indistintamente S o U, deberá asegurarse de que empleando S no afectará la operación del procesador. Es, pues,

más seguro el empleo de U. Estos dos registros de pila hacen del 6809 un procesador ideal con FORTH.

- La utilización de los registros X, Y, S y U no tiene por qué limitarse a sus destinos originales: tanto S como U pueden servir de registros índice, por ejemplo, y los cuatro registros pueden emplearse para almacenar y manipular números de 16 bits.
- El contador de programa (PC) es un registro de 16 bits que se ajusta automáticamente señalando la instrucción siguiente a ejecutar. Las instrucciones JMP (salto) y BRA (bifurcación) alteran su contenido y el 6809 permite que sea usado como una especie de registro índice. Ya analizaremos más adelante los posibles efectos; aquí basta saber que la utilización de direcciones relativas al contenido del PC, en vez de direcciones absolutas almacenadas en X o Y, brinda al programador un código independiente de la posición. En otras palabras, los programas escritos adecuadamente pueden cargarse en cualquier posición de la memoria y ser ejecutados desde allí.
- El registro de código de condición (CC) tiene ocho bits, empleados por separado como flags indicativos de la condición del procesador. Una instrucción como BNE (*branch not equal*: bifurcar si no es igual) consulta uno de estos flags, causando un cambio en el flujo del control según la condición del flag (cero o uno).

Existen diversas instrucciones destinadas al traslado de los datos dentro, fuera o entre los diversos registros. Sirvan los siguientes ejemplos. Supongamos que tenemos reservadas algunas posiciones de memoria mediante las directivas de assembly FCB y FDB. El procesador no las toma como instrucciones. En el momento de traducir el programa assembly a código máquina, las obedecerá inmediatamente reservando las posiciones deseadas para uso del programa. El ensamblador no traducirá las directivas a código máquina porque no tienen interés para el procesador. Dado que se parecen a códigos de operación en assembly (del tipo BNE o JMP) se suelen llamar *pseudo-ops*; pero es preferible el nombre de *directivas de assembler*, que explica mejor su función: dirigir el funcionamiento del programa assembler. En el caso de que tales directivas estén etique-

Un trío amistoso

El muy avanzado microprocesador 6809 de ocho bits de Motorola se relaciona con el "viejo y siempre fiel" Mostech 6502 a través de su ancestro común, el Motorola 6800. Esta relación es transparente si se examina sus lenguajes assembly tan parecidos, pero la estructura de registros del 6809, con sus registros índice de 16 bits y su pareja de acumuladores AB, se acerca más a la del Zilog Z80, proveniente a su vez del Intel. Los dos registros de pila y el registro de página directa son exclusivos del 6809, aunque ese último registro es una variante avanzada de la poderosa facilidad que tiene el 6502 para el direccionamiento con página cero. Aunque es más sólido técnicamente que sus competidores, la tardía distribución del 6809 y la llegada de los procesadores de 16 bits han oscurecido mucho su posición en el mercado

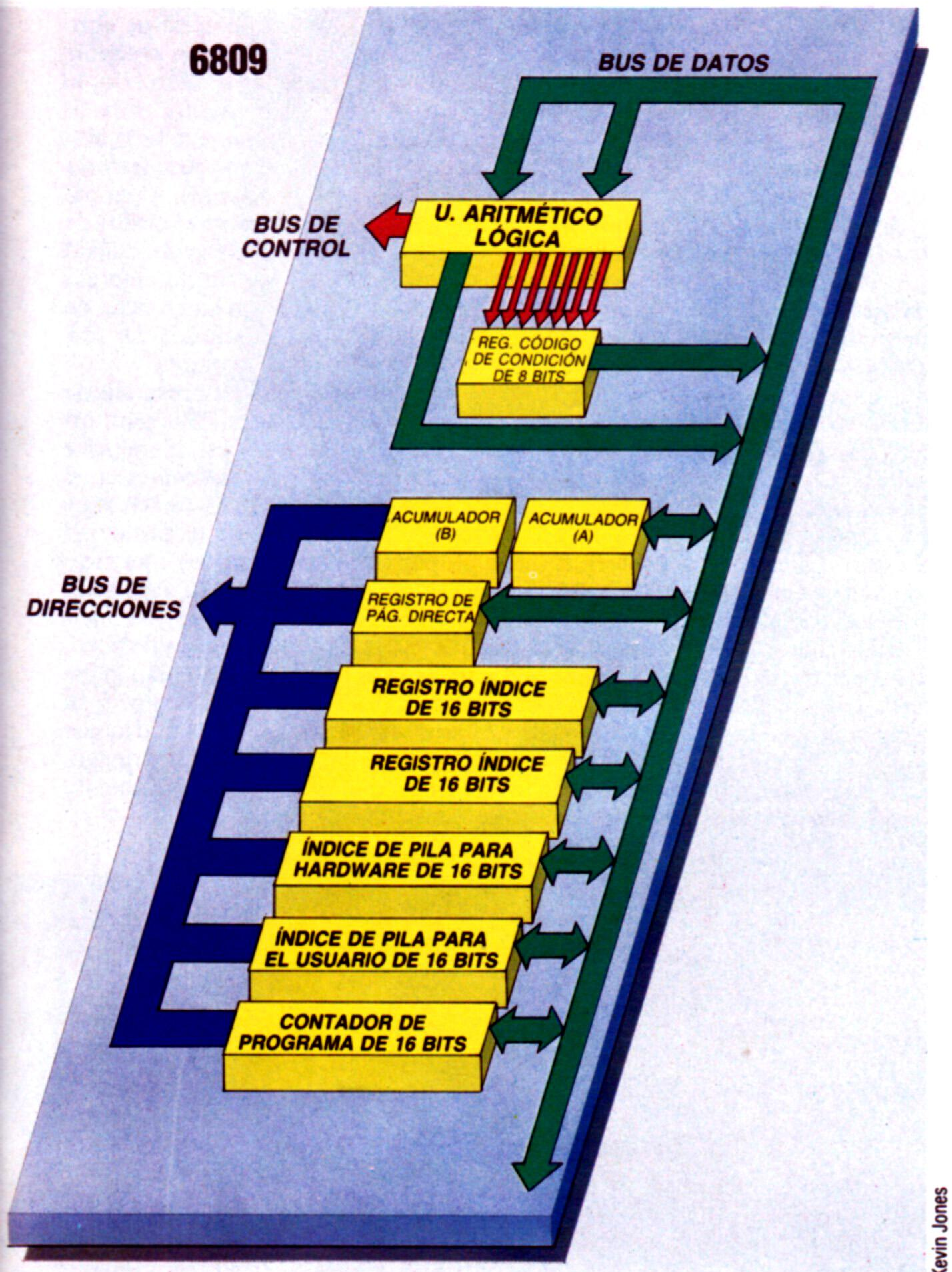


tadas, el assembler traducirá la etiqueta en la dirección apropiada, como en este ejemplo:

NUM1 FCB 0	reservar un byte que se denominará NUM1, con valor inicial 0
NUM2 FCB 0	similar al anterior pero etiquetado con otro nombre
NUM3 FDB #A93B	reserva dos bytes para el número de 16 bits #A93B (el símbolo # se utiliza para indicar que un número está escrito en notación hexadecimal)

Las siguientes instrucciones tienen por finalidad cargar los valores almacenados en tales posiciones dentro de los diversos registros (LD; del inglés *load*: cargar):

LDA NUM1	cargará en el acumulador el número de ocho bits almacenado en la posición de memoria referenciada por NUM1
LDB NUM2	carga, como antes, en el acumulador B el contenido de NUM2



Kevin Jones

```
LDX NUM3
LDY NUM3
LDS NUM3
LDU NUM3
LDD NUM3
```

instrucciones que cargarán el número de 16 bits contenido en NUM3 dentro de los registros X,Y,S,U y D respectivamente

De forma similar, el contenido, de 8 o 16 bits, de cualquier registro puede almacenarse en una posición de memoria por medio de una de estas instrucciones (ST; del inglés *store*: almacenar):

```
STA NUM1
STB NUM2
STX NUM3
STY NUM3
STS NUM3
STU NUM3
STD NUM3
```

Adviértase que cuando el acumulador es cargado (LD) con el valor de NUM1 el contenido de la posición de memoria NUM1 no se altera; de forma similar ocurre con las operaciones de almacenamiento (ST).

Se pueden intercambiar *(exchange)* contenidos de dos registros (siempre que ambos sean del mismo tamaño) mediante la instrucción EXG. Por ejemplo:

EXG A,B	lo que contiene A se traslada a B y viceversa
EXG X,S	intercambio de contenidos entre X y S

Lo que contiene un registro se puede trasladar *(transfer)* a otro registro. Por ejemplo: TFR Y,U copia en U lo que contiene Y. De nuevo es condición indispensable que ambos registros sean del mismo tamaño (8 a 16 bits).

Para terminar esta lección, vamos a presentar una instrucción que realiza una tarea concreta y que nos permitirá escribir un sencillo programa. La instrucción ADD (en inglés, "sumar") le suma al valor contenido en el acumulador un valor contenido en una posición de memoria (recuérdese que disponemos de dos acumuladores):

ADDA NUM1	significa: "suma el contenido de la posición NUM1 al contenido del registro A y el resultado déjalo en dicho registro A"

Vamos primero a sumar los dos números de ocho bits que están en NUM1 y NUM2, dejando el resultado en NUM1 y descuidando la posibilidad de un desbordamiento si la suma supera un número de ocho bits. Sumaremos después los contenidos de las dos posiciones, pero con la obtención de un resultado de 16 bits que se dejará en NUM3.

Primer ejemplo:

LDA NUM1	carga el primer número dentro de A
ADDA NUM2	agrega a A el segundo número
STA NUM1	almacena el resultado en NUM1

Segundo ejemplo:

LDB NUM1	carga el primer número en B
SEX	convierte el número anterior en un número de 16 bits cargándolo en D
STD NUM3	almacena D en NUM3
LDB NUM2	carga el segundo número en B
SEX	conversión a 16 bits y carga en D
ADDD NUM3	suma a D el contenido de NUM3, que es un número de 16 bits
STD NUM3	almacena el resultado en NUM3

Cuidando la imagen

La creatividad australiana se halla detrás de la alta calidad del software producido por la empresa Melbourne House

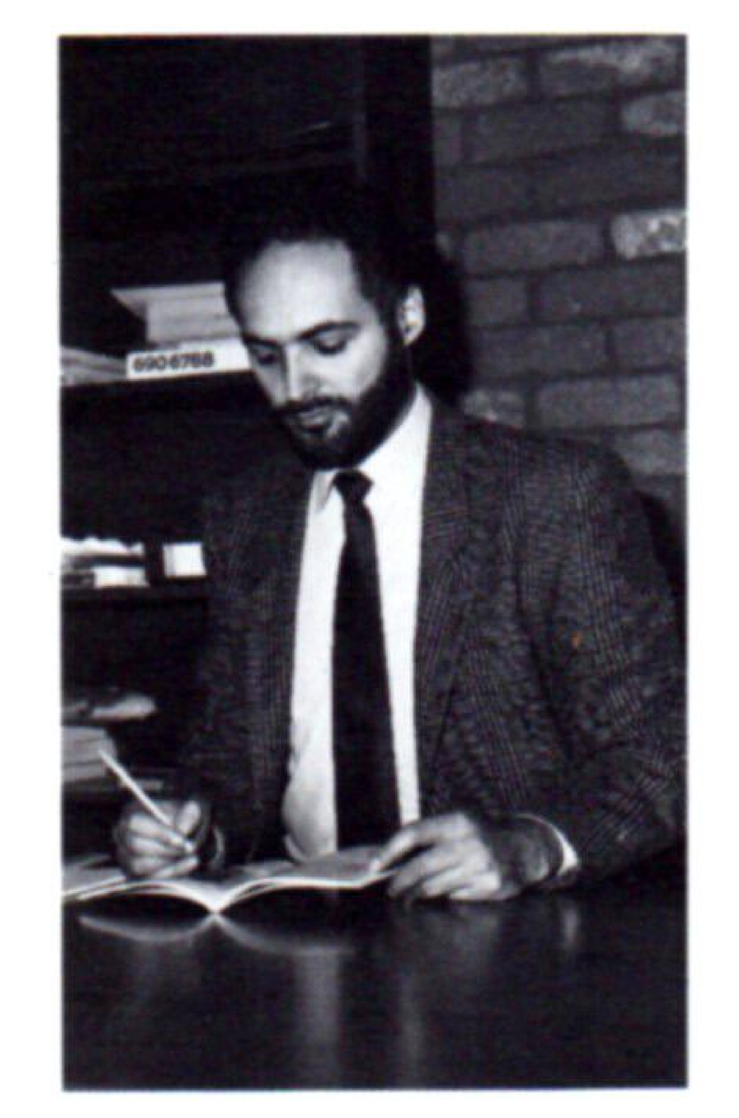
Melbourne House fue fundada en 1977 por el australiano Fred Milgrom. El lanzamiento del Sinclair ZX80 alertó a Milgrom de los potenciales beneficios de editar libros sobre la informática personal, y en 1980 Melbourne House produjo *30 Programs for the ZX80 (Treinta programas para el ZX80)*. El éxito de esta publicación fue el origen de una serie de libros dedicados a la máquina Sinclair, y la empresa sacó al mercado su primer paquete de software: *Space invaders (Invasores del espacio)*, nuevamente para el ZX80.

El año siguiente vio la aparición del ZX81. La demanda de libros y cassettes para el ZX80 cayeron a plomo, y fueron las ventas en Estados Unidos las que salvaron a la empresa del desastre. La lección se aprendió bien y Melbourne House comprendió las ventajas de la diversificación. A medida que iban saliendo al mercado nuevas máquinas, la empresa surtía a los usuarios de libros y software, cuyas ventas se veían favorecidas por la baja calidad de las guías para el usuario que se suministraban con algunos ordenadores personales.

El éxito inmediato del Sinclair Spectrum facilitó que Melbourne House produjera software de juegos explotando los gráficos en color en alta resolución y el sonido de la máquina. El juego recreativo *Penetrator* se vendió bien, pero el mayor éxito de la empresa fue el juego *The Hobbit*, una aventura gráfica basada en la novela homónima de J.R.R. Tolkien, que ganó el premio Golden Joystick al mejor juego de estrategia del año. La cassette del juego se comercializó en un paquete que también contenía un ejemplar del libro de Tolkien; ésta fue una condición que impusieron los albaceas de los bienes del autor y determinó que *The Hobbit* se vendiera a un precio tres veces superior al de la mayoría de los paquetes de software que se editaban para el Spectrum en aquellos tiempos. A pesar del costo, las ventas fueron sumamente buenas y ahora *The Hobbit* se comercializa también para otras máquinas, incluyendo el BBC Micro y el Oric/Atmos.

Los programadores de la empresa están afincados en Melbourne (Australia). Cada equipo, compuesto por cuatro miembros, se concentra en un aspecto de un juego. Ello significa que se invierte más tiempo en el desarrollo de los juegos, pero esta política les ha valido enormes ventas y la lealtad de los clientes. La empresa espera capitalizar esta fidelidad en una campaña para vender más libros. Paula Byrne, directora de publicidad de la empresa, señala: "Cuando la gente va a comprar un libro no sabe bien lo que desea y, por tanto, es probable que acabe adquiriendo algo que no es adecuado, lo que les hace perder el interés por adquirir otros libros". Melbourne House confía en combatir esta confusión clasificando sus libros como para lectores principiantes, intermedios o avanzados, y empaquetando libros y software en el mismo estilo, de modo que el público identifique libros de calidad con software de calidad. Con este fin, la empresa incluye en cada uno de sus productos una ficha de registro en la que se invita a los clientes a dar una opinión acerca de la calidad de su compra.

De su último juego, *Mugsy*, la empresa afirma que es "la primera tira cómica interactiva para ordenador del mundo", y permite que el jugador asuma el control de una banda de maleantes en el Chicago de los años veinte. Los gráficos son muy detallados y responden a un diseño fantástico, si bien el juego propiamente dicho es poco complicado. Melbourne House está trabajando, asimismo, en una aventura "estilo *Hobbit*" denominada *Sherlock Holmes*, que saldrá a la venta dentro de poco. Se dice que este juego es tan innovador como lo fue *The Hobbit* en la época de su lanzamiento, y su desarrollo ha llevado 15 meses. Es muy poco lo que ha trascendido acerca del contenido del juego, ¡aunque se dice que requiere un buen conocimiento del transporte victoriano!

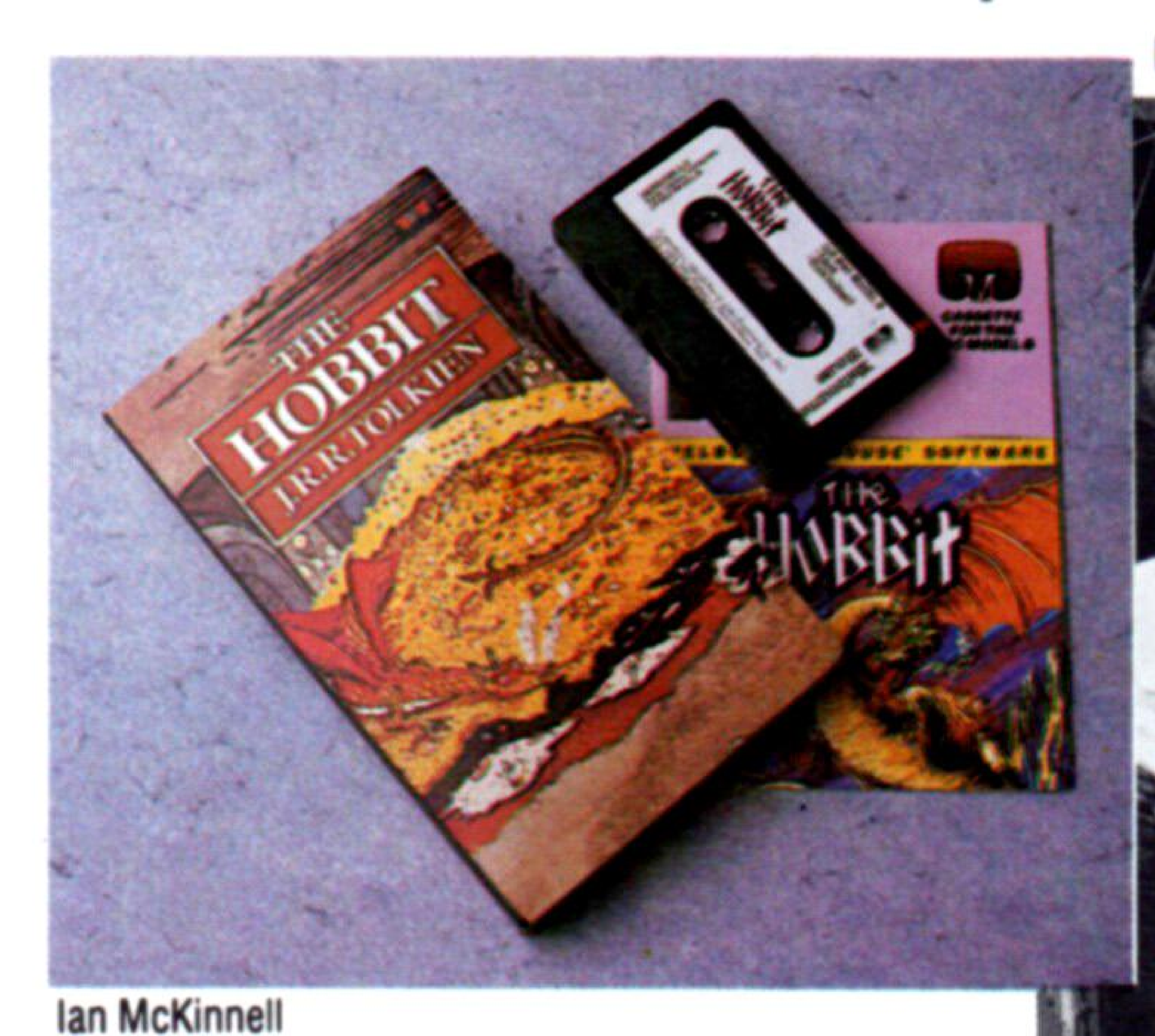


Alfred Milgrom, codirector y editor de Melbourne House



Philip Mitchell, autor del juego "The Hobbit" y de "Sherlock Holmes"

El libro del juego
Con *The Hobbit*, además de crear un juego divertido y emocionante, Melbourne House también incluye un ejemplar de la obra maestra de J.R.R. Tolkien; esta iniciativa ha constituido una brillante táctica de ventas



Ian McKinnell

Equipo de software de Melbourne House



mi COMPUTER

Editorial Delta, S.A.



10051
9 788485 822836